```
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
```

```
Va
--
00
00
00
00
00
00
00
00
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
```

FILEID**EDFUTIL

```
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF  UU      UU  TTTTTTTTTT    IIIIII    LL
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF  UU      UU  TTTTTTTTTT    IIIIII    LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EEEEEEEE    DD      DD  FFFFFFFF    UU      UU      TT          II      LL
EEEEEEEE    DD      DD  FFFFFFFF    UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL
EE          DD      DD  FF          UU      UU      TT          II      LL              ....
EE          DD      DD  FF          UU      UU      TT          II      LL              ....
EEEEEEEEEE  DDDDDDDD    FF          UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL      ....
EEEEEEEEEE  DDDDDDDD    FF          UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL      ....


LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

0000000000

```
0001            [ IDENT ('V04-000'),
0002        { ++
0003        ******************************************************************************
0004        **                                                                          *
0005        **  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0006        **  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0007        **  ALL RIGHTS RESERVED.                                                    *
0008        **                                                                          *
0009        **  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0010        **  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0011        **  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0012        **  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0013        **  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0014        **  TRANSFERRED.                                                            *
0015        **                                                                          *
0016        **  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0017        **  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0018        **  CORPORATION.                                                            *
0019        **                                                                          *
0020        **  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0021        **  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0022        **                                                                          *
0023        **                                                                          *
0024        ******************************************************************************
0025
0026
0027
0028
0029
0030        FACILITY:       VAX/VMS EDF (EDIT/FDL) UTILITY
0031
0032        ABSTRACT:       This facility is used to create, modify, and optimize
0033                        FDL specification files.
0034
0035        ENVIRONMENT:    NATIVE/USER MODE
0036
0037        AUTHOR:         Ken F. Henderson Jr.
0038
0039        CREATION DATE:  27-Mar-1981
0040
0041        MODIFIED BY:
0042
0043                        V03-012 RRB0005         Rowland R. Bradley      16 Jan 1984
0044                                Fix EDF$RESET_SCROLL to avoid overwriting the output
0045                                generated by a command procedure with SET VERIFY set.
0046
0047                        V03-011 RRB0004         Rowland R. Bradley      13 Jan 1984
0048                                Fix MAX_FACTOR to prevent division by zero.
0049
0050                        V03-010 KFH0010         Ken Henderson           10 Sep 1983
0051                                Support for named UICs.
0052
0053                        V03-009 KFH0009         Ken Henderson            8 Aug 1983
0054                                Changes for seperate compilation.
0055
0056                        V03-008 KFH0008         Ken Henderson           28 Jul 1983
0057                                Added CALC_REC_OVERHEAD and CALC_BUC_OVERHEAD
```

```
0058                    to centralize the arithmetic.
0059                    Fixed EDF$LINE_PARSED to not invert the
0060                    order of block comment lines.
0061
0062            V03-007 KFH0007         Ken Henderson           26 Apr 1983
0063                    Fix NUMBER_INPUT to set INPUT_VALUE
0064                    and INPUT_NUMBER also. Modify
0065                    SCAN_DEFINITION routine to accept
0066                    FATAL parameter. Modify CURRENT_LT_TEST
0067                    and CURRENT_GT_TEST routines to
0068                    reverse precedence of SECONDARY
0069                    and SECNUM
0070                    - and make SEG_TYPE SECNUM = 7.
0071
0072            V03-006 KFH0006         Ken Henderson           14 Apr 1983
0073                    Added support for JOURNAL_ENABLED.
0074                    Added MAX_FACTOR, DELETE_PRIMARY_SECTION
0075                    routines.
0076
0077            V03-005 KFH0005         Ken Henderson           31 Jan 1983
0078                    Added XAB$C_BN8 and XAB$C_IN8 to
0079                    EDF$LINE_PARSED. And changed the
0080                    reference of FDL$TYPE to FDL3$TYPE.
0081
0082            V03-004 KFH0004         Ken Henderson           11 Jan 1983
0083                    Modified EDF$RESET_SCROLL to say
0084                    "Created:" in reverse video.
0085
0086            V03-003 KFH0003         Ken Henderson           8 Sept 1982
0087                    Modified reference to some variables
0088                    to fit with database reorganization.
0089                    Also, modified call to ASK_RETURN.
0090
0091            V03-002 KFH0002         Ken Henderson           2 April 1982
0092                    Modified INSERT_IN_ORDER to not
0093                    start at DEF_HEAD if it was already
0094                    at the correct place.
0095
0096            V03-001 KFH0001         Ken Henderson           23-Mar-1982
0097                    Modified EDF$RESET_SCROLL to not
0098                    reset the scrolling region unless
0099                    it has been set.
0100
0101        -- }
```

```
0103            ENVIRONMENT ('LIB$:EDFUTIL'),
0104
0105            INHERIT (
0106
0107            'SYS$LIBRARY:STARLET',
0108            'SHRLIB$:FDLPARDEF',
0109            'LIB$:EDFSDLMSG',
0110            'LIB$:EDFSTRUCT',
0111            'LIB$:EDFCONST',
0112            'LIB$:EDFTYPE',
0113            'LIB$:EDFVAR',
0114            'LIB$:EDFEXTERN',
0115            'LIB$:EDFCHF'
0116
0117            )]
0118
0119            MODULE EDFUTIL (INPUT,OUTPUT);
```

```
0121            { ++
0122
0123            NUM_LEN -- Calculate the field width of an integer.
0124
0125            This routine will return the number of characters an integer will take up
0126            when printed.
0127
0128            CALLING SEQUENCE:
0129
0130            field-width     := NUM_LEN (NUMBER);
0131
0132            INPUT PARAMETERS:
0133
0134            NUMBER
0135
0136            IMPLICIT INPUTS:
0137
0138            none
0139
0140            OUTPUT PARAMETERS:
0141
0142            none
0143
0144            IMPLICIT OUTPUTS:
0145
0146            none
0147
0148            ROUTINES CALLED:
0149
0150            none
0151
0152            ROUTINE VALUE:
0153
0154            The field width
0155
0156            SIGNALS:
0157
0158            none
0159
0160            SIDE EFFECTS:
0161
0162            none
0163
0164            -- }
```

```
0166            [ASYNCHRONOUS] FUNCTION NUM_LEN (
0167                                NUMBER      : INTEGER
0168                                ) : INTEGER;
0169
0170            VAR
0171                TEST_VAR    : INTEGER;
0172                TEST_LEN    : INTEGER;
0173
0174            BEGIN
0175
0176                IF NUMBER = 0 THEN
0177
0178                    { +
0179                    Just plug a width of 1 if the number is 0.
0180                    - }
0181                    NUM_LEN                 := 1
0182
0183                ELSE
0184
0185                BEGIN
0186
0187                    { +
0188                    Set the function value according to the magnitude of the number.
0189                    - }
0190                    TEST_VAR                := 1000000000;
0191                    TEST_LEN                := 10;
0192
0193                    REPEAT
0194
0195                        IF ABS (NUMBER) < TEST_VAR THEN
0196
0197                            TEST_LEN        := TEST_LEN - 1;
0198
0199                        TEST_VAR            := TEST_VAR DIV 10;
0200
0201                    UNTIL ABS (NUMBER) >= TEST_VAR;
0202
0203                    { +
0204                    Allow for a - sign if negative.
0205                    - }
0206                    IF NUMBER < 0 THEN
0207
0208                        TEST_LEN            := TEST_LEN + 1;
0209
0210                    { +
0211                    Now stuff the function value.
0212                    - }
0213                    NUM_LEN                 := TEST_LEN;
0214
0215                END;        { IF FALSE NUMBER = 0 }
0216
0217            END;    { NUM_LEN }
```

```
0219            { ++
0220
0221            MAX_FACTOR -- Produce a number that's a multiple of another.
0222
0223            This function will return the number that's a multiple of one of the
0224            arguments, as long as it doesn't go over a maximum.
0225
0226            CALLING SEQUENCE:
0227
0228            NEWVALUE := MAX_FACTOR (BASE,VALUE,MAX);
0229
0230            INPUT PARAMETERS:
0231
0232            BASE
0233            VALUE
0234            MAX
0235
0236            IMPLICIT INPUTS:
0237
0238            none
0239
0240            OUTPUT PARAMETERS:
0241
0242            none
0243
0244            IMPLICIT OUTPUTS:
0245
0246            none
0247
0248            ROUTINES CALLED:
0249
0250            none
0251
0252            ROUTINE VALUE:
0253
0254            The number (greater or equal to VALUE) that's a multiple of BASE,
0255            unless that number would be greater than MAX - in which case it is MAX.
0256
0257            SIGNALS:
0258
0259            none
0260
0261            SIDE EFFECTS:
0262
0263            none
0264
0265            -- }
```

```
0267            FUNCTION MAX_FACTOR (
0268                                BASE        : INTEGER;
0269                                VALUE       : INTEGER;
0270                                MAX         : INTEGER
0271                                ) : INTEGER;
0272
0273            VAR
0274                TEMP        : INTEGER;
0275
0276            BEGIN
0277
0278                IF (VALUE < BASE) OR (BASE = 0) THEN
0279
0280                    TEMP    := BASE
0281
0282                ELSE
0283
0284                BEGIN
0285
0286                    TEMP    := VALUE DIV BASE;
0287
0288                    IF ((VALUE MOD BASE) <> 0) THEN
0289
0290                        TEMP        := TEMP + 1;
0291
0292                    TEMP    := TEMP * BASE;
0293
0294                END;
0295
0296                IF TEMP > MAX THEN
0297
0298                    TEMP    := MAX;
0299
0300                MAX_FACTOR  := TEMP;
0301
0302            END;    { MAX_FACTOR }
```

```
0304            { ++
0305
0306            CALC_REC_OVERHEAD -- Do the arithmetic to figure out overheads.
0307
0308            This function will return the RELORD overhead for a given setup.
0309
0310            CALLING SEQUENCE:
0311
0312            RECORD_OVERHEAD := CALC_REC_OVERHEAD (INDEX_LEVEL);
0313
0314            INPUT PARAMETERS:
0315
0316            INDEX_LEVEL
0317
0318            IMPLICIT INPUTS:
0319
0320            none
0321
0322            OUTPUT PARAMETERS:
0323
0324            none
0325
0326            IMPLICIT OUTPUTS:
0327
0328            none
0329
0330            ROUTINES CALLED:
0331
0332            none
0333
0334            ROUTINE VALUE:
0335
0336            The overhead, according to the RMS structure constants.
0337
0338            SIGNALS:
0339
0340            none
0341
0342            SIDE EFFECTS:
0343
0344            none
0345
0346            -- }
```

```
0348        FUNCTION CALC_REC_OVERHEAD (
0349                            INDEX_LEVEL : INTEGER
0350                            ) : INTEGER;
0351
0352        VAR
0353            RECORD_OVERHEAD : INTEGER;
0354
0355        BEGIN
0356
0357            RECORD_OVERHEAD     := 0;
0358
0359            { +
0360            SIDR BUCKET
0361            - }
0362            IF (IDATA[EDF$K_ACTIVE_KEY] <> 0) AND (INDEX_LEVEL = 0) THEN
0363
0364                RECORD_OVERHEAD := RECORD_OVERHEAD + IRC$C_SDROVHSZ3 + IRC$C_RRVOVHSZ3;
0365
0366            { +
0367            ACCOUNT FOR KEY COMPRESSION
0368            - }
0369            IF (
0370            (BDATA[EDF$K_KEY_COMP_WANTED] AND (INDEX_LEVEL = 0))
0371            OR
0372            (BDATA[EDF$K_IDX_COMP_WANTED] AND (INDEX_LEVEL <> 0))
0373            ) THEN
0374
0375                RECORD_OVERHEAD := RECORD_OVERHEAD + IRC$C_KEYCMPOVH;
0376
0377            { +
0378            INDEX BUCKETS
0379            - }
0380            IF INDEX_LEVEL <> 0 THEN
0381
0382                RECORD_OVERHEAD := RECORD_OVERHEAD + IRC$C_MAXVBNSZ;
0383
0384            { +
0385            PRIMARY KEY DATA BUCKETS
0386            - }
0387            IF (IDATA[EDF$K_ACTIVE_KEY] = 0) AND (INDEX_LEVEL = 0) THEN
0388
0389                BEGIN
0390
0391                IF VARIABLE_RECORDS THEN
0392
0393                    RECORD_OVERHEAD     := RECORD_OVERHEAD + IRC$C_VAROVHSZ3
0394
0395                ELSE
0396
0397                    RECORD_OVERHEAD     := RECORD_OVERHEAD + IRC$C_FIXOVHSZ3;
0398
0399                IF BDATA[EDF$K_REC_COMP_WANTED] THEN
0400
0401                    RECORD_OVERHEAD     := RECORD_OVERHEAD + IRC$C_DATCMPOVH;
0402
0403                IF BDATA[EDF$K_KEY_COMP_WANTED] OR BDATA[EDF$K_REC_COMP_WANTED] THEN
0404
```

```
0405                    RECORD_OVERHEAD     := RECORD_OVERHEAD + IDATA[EDF$K_KEY_SIZE];
0406
0407                END;
0408
0409            CALC_REC_OVERHEAD   := RECORD_OVERHEAD;
0410
0411        END;    { CALC_REC_OVERHEAD }
```

```
0413            { ++
0414
0415            CALC_BUC_OVERHEAD -- Do the arithmetic to figure out overheads.
0416
0417            This function will return the BUCKET overhead for a given setup.
0418
0419            CALLING SEQUENCE:
0420
0421            BUCKET_OVERHEAD := CALC_BUC_OVERHEAD (INDEX_LEVEL);
0422
0423            INPUT PARAMETERS:
0424
0425            INDEX_LEVEL
0426
0427            IMPLICIT INPUTS:
0428
0429            none
0430
0431            OUTPUT PARAMETERS:
0432
0433            none
0434
0435            IMPLICIT OUTPUTS:
0436
0437            none
0438
0439            ROUTINES CALLED:
0440
0441            none
0442
0443            ROUTINE VALUE:
0444
0445            The overhead, according to the RMS structure constants.
0446
0447            SIGNALS:
0448
0449            none
0450
0451            SIDE EFFECTS:
0452
0453            none
0454
0455            -- }
```

```
0457        FUNCTION CALC_BUC_OVERHEAD (
0458                             INDEX_LEVEL : INTEGER
0459                             ) : INTEGER;
0460
0461        BEGIN
0462
0463            IF INDEX_LEVEL = 0 THEN
0464
0465                CALC_BUC_OVERHEAD       := BKT$C_OVERHDSZ + BKT$C_DATBKTOVH
0466
0467            ELSE
0468
0469                CALC_BUC_OVERHEAD       := BKT$C_OVERHDSZ + BKT$C_ENDOVHD;
0470
0471        END;    { CALC_BUC_OVERHEAD }
```

```
0473            { ++
0474
0475            EDF$RESET_SCROLL -- Reset an ANSI terminal's scroll region.
0476
0477            This routine will put the scroll region back to full screen.
0478            It also clears graphics mode.
0479            It is a Global routine, which is called by the exit handler as well.
0480
0481            CALLING SEQUENCE:
0482
0483            EDF$RESET_SCROLL;
0484
0485            INPUT PARAMETERS:
0486
0487            none
0488
0489            IMPLICIT INPUTS:
0490
0491            LINE_ONE
0492            LINES_PER_PAGE
0493
0494            OUTPUT PARAMETERS:
0495
0496            none
0497
0498            IMPLICIT OUTPUTS:
0499
0500            SYS$OUTPUT: - the scroll region is reset, and possibly graphics mode reset
0501
0502            ROUTINES CALLED:
0503
0504            LIB$SET_SCROLL
0505
0506            ROUTINE VALUE:
0507
0508            none
0509
0510            SIGNALS:
0511
0512            none
0513
0514            SIDE EFFECTS:
0515
0516            none
0517
0518            -- }
```

```
0520            [ASYNCHRONOUS,GLOBAL] PROCEDURE EDF$RESET_SCROLL;
0521
0522            BEGIN
0523
0524                IF NOT AUTO_TUNE THEN
0525
0526                BEGIN
0527
0528                    { +
0529                    Clear graphics mode if this is a Regis device.
0530                    - }
0531                    IF REGIS THEN
0532
0533                    BEGIN
0534
0535                        CHFFLAGS    := 0;
0536                        WRITEV (OUT_LINE,''(27)'\');
0537                        LIB$PUT_LINE (OUT_LINE,ONE,CHFFLAGS);
0538
0539                    END;
0540
0541                    { +
0542                    Now make the scroll region from top to bottom - if it was ever set.
0543                    - }
0544                    IF SCROLLING_SET THEN
0545
0546                        LIB$SET_SCROLL (LINE_ONE,LINES_PER_PAGE);
0547
0548                END;        { IF NOT AUTO_TUNE }
0549
0550                { +
0551                Announce that the file has been created.
0552                - }
0553                IF (
0554                (FILE_CREATED)
0555                AND
0556                (RES_OUTPUT_FILENAME_DESC.DSC$W_LENGTH > 0)
0557                ) THEN
0558
0559                BEGIN
0560
0561                    CHFFLAGS        := SCR$M_REVERSE;
0562                    WRITEV (OUT_LINE,CRLF,
0563            RES_OUTPUT_FILENAME_DESC.DSC$A_POINTER^:RES_OUTPUT_FILENAME_DESC.DSC$W_LENGTH,
0564            '  ',LINES_SHOWN:NUM_LEN(LINES_SHOWN),' lines');
0565                    LIB$PUT_LINE (OUT_LINE,ONE,CHFFLAGS);
0566
0567                END;
0568
0569            END;    { EDF$RESET_SCROLL }
```

```
0571            { ++
0572
0573            CLEAR -- Clear a designated area of the screen.
0574
0575            This routine clears a specific area on the screen and leaves the cursor there.
0576            It bypasses screwing up non-CRT terminals.
0577
0578            CALLING SEQUENCE:
0579
0580            CLEAR (DESTINATION);
0581
0582            INPUT PARAMETERS:
0583
0584            DESTINATION
0585
0586            IMPLICIT INPUTS:
0587
0588            PROMPT_LINE
0589            LINE_ONE
0590
0591            OUTPUT PARAMETERS:
0592
0593            none
0594
0595            IMPLICIT OUTPUTS:
0596
0597            SYS$OUTPUT:
0598
0599            ROUTINES CALLED:
0600
0601            LIB$ERASE_PAGE
0602            LIB$ERASE_LINE
0603
0604            ROUTINE VALUE:
0605
0606            none
0607
0608            SIGNALS:
0609
0610
0611            SIDE EFFECTS:
0612
0613            The selected lines on the screen are cleared (unless hardcopy).
0614
0615            -- }
```

```
0617            PROCEDURE CLEAR (
0618                            DESTINATION : INTEGER
0619                            );
0620
0621            BEGIN
0622
0623                {+
0624                All this stuff affects only video terminals.
0625                -}
0626                IF (
0627                (VIDEO_TERMINAL)
0628                AND
0629                (NOT AUTO_TUNE)
0630                ) THEN
0631
0632                BEGIN
0633
0634                    CASE DESTINATION OF
0635
0636                        SCREEN :
0637
0638                            BEGIN
0639
0640                                {+
0641                                The following sequence of junk to the screen
0642                                is overkill to make sure the titles don't
0643                                jump around. (interaction of Pascal I/O and
0644                                screen package I/O...)
0645                                -}
0646                                IF REGIS THEN
0647
0648                                    WRITELN (''(27)'Pp;S(E);'(27)'\');
0649
0650                                LIB$ERASE_PAGE (LINE_ONE,COL_ONE);
0651                                WRITELN (' ');
0652                                LIB$SET_CURSOR (LINE_ONE,COL_ONE);
0653
0654                            END;    { SCREEN }
0655
0656                        LOWER_AREA :
0657
0658                        BEGIN
0659
0660                            IF REGIS THEN
0661
0662                            BEGIN
0663
0664                                WRITELN (
0665                                ''(27)'PpP[27,320];V(W(I0,S1,E,S[,479]))[+767];'(27)'\');
0666
0667                                LIB$SET_CURSOR (PROMPT_LINE,COL_ONE);
0668
0669                            END
0670
0671                            ELSE
0672
0673                                LIB$ERASE_PAGE (LOWER_LINE,COL_ONE);
```

```
0674
0675                END;
0676
0677                IF_FULL_PROMPT :    IF FULL_PROMPT OR TEMP_FULL_PROMPT THEN
0678
0679                    BEGIN
0680
0681                        IF TEMP_FULL_PROMPT THEN
0682
0683                            LIB$WAIT (1.3);
0684
0685                        { +
0686                        The following sequence of junk to the screen
0687                        is overkill to make sure the titles don't
0688                        jump around. (interaction of Pascal I/O and
0689                        screen package I/O...)
0690                        - }
0691
0692                        IF REGIS THEN
0693
0694                            WRITELN (''(27)'Pp;S(E);'(27)'\');
0695
0696                        LIB$ERASE_PAGE (LINE_ONE,COL_ONE);
0697                        WRITELN (' ');
0698                        LIB$SET_CURSOR (LINE_ONE,COL_ONE);
0699
0700                    END;    { IF_FULL_PROMPT }
0701
0702                PAUSE :             QUERY (EDF$K_RETURN);
0703
0704                OTHERWISE
0705
0706                    { NULL-STATMENT } ;
0707
0708                END;            { CASE }
0709
0710            END;        { IF VIDEO_TERMINAL AND NOT AUTO_TUNE }
0711
0712        END;    { CLEAR }
```

```
0714            { ++
0715
0716            CVT_QUAD_DESC -- Routine to convert a quadword to a descriptor.
0717
0718            This routine will take 2 longword arguments and stuff them into a descriptor.
0719
0720            CALLING SEQUENCE:
0721
0722            DESCRIPTOR_VAR  := CVT_QUAD_DESC (LONG1,LONG2);
0723
0724            INPUT PARAMETERS:
0725
0726            LONG1
0727            LONG2
0728
0729            IMPLICIT INPUTS:
0730
0731            none
0732
0733            OUTPUT PARAMETERS:
0734
0735            none
0736
0737            IMPLICIT OUTPUTS:
0738
0739            none
0740
0741            ROUTINES CALLED:
0742
0743            none
0744
0745            ROUTINE VALUE:
0746
0747            DESCRIPTOR_VAR
0748
0749            SIGNALS:
0750
0751            none
0752
0753            SIDE EFFECTS:
0754
0755            none
0756
0757            -- }
```

```
0759            [ASYNCHRONOUS] FUNCTION CVT_QUAD_DESC (LONG1,LONG2 : LONG) : DESCRIPTOR;
0760
0761            BEGIN
0762
0763                WITH QUAD_DESC DO
0764
0765                BEGIN
0766
0767                    { +
0768                    Select the quadword type variant and stuff it.
0769                    - }
0770                    QWHICH                  := QWORD;
0771                    TWOLONG.L1              := LONG1;
0772                    TWOLONG.L2              := LONG2;
0773
0774                    { +
0775                    Now select the descriptor type variant and get it.
0776                    - }
0777                    QWHICH                  := DWORD;
0778                    CVT_QUAD_DESC           := DSC;
0779
0780                END;
0781
0782            END;    { CVT_QUAD_DESC }
```

```
0784            { ++
0785
0786            SCAN_DEFINITION -- Search for area and key primaries in the current definition
0787            and log them.
0788
0789            CALLING SEQUENCE:
0790
0791            SCAN_DEFINITION (FATAL);
0792
0793            INPUT PARAMETERS:
0794
0795            FATAL
0796
0797            IMPLICIT INPUTS:
0798
0799            DEF_CURRENT
0800
0801            OUTPUT PARAMETERS:
0802
0803            none
0804
0805            IMPLICIT OUTPUTS:
0806
0807            LOW_AREA
0808            HIGH_AREA
0809            LOW_KEY
0810            HIGH_KEY
0811            FOUND_0
0812            FOUND_AREA
0813            FOUND_KEY
0814
0815            ROUTINES CALLED:
0816
0817            none
0818
0819            ROUTINE VALUE:
0820
0821            none
0822
0823            SIGNALS:
0824
0825            none
0826
0827            SIDE EFFECTS:
0828
0829            none
0830
0831            -- }
```

```
0833        PROCEDURE SCAN_DEFINITION (FATAL : BOOLEAN);
0834
0835        BEGIN
0836
0837            { *
0838            Find out the range of existing keys (assume contiguous).
0839            * }
0840            DEF_CURRENT         := DEF_HEAD;
0841            FOUND_0             := FALSE;
0842            FOUND_AREA          := FALSE;
0843            FOUND_KEY           := FALSE;
0844            LOW_AREA            := 0;
0845            HIGH_AREA           := 0;
0846            LOW_KEY             := 0;
0847            HIGH_KEY            := 0;
0848
0849            REPEAT
0850
0851                WITH DEF_CURRENT^ DO
0852
0853                BEGIN
0854
0855                    IF (
0856                    (OBJECT_TYPE = PRI)
0857                    AND
0858                    (PRIMARY = KEY)
0859                    ) THEN
0860
0861                    BEGIN
0862
0863                        IF PRINUM = 0 THEN
0864
0865                            FOUND_0     := TRUE;
0866
0867                        FOUND_KEY       := TRUE;
0868
0869                        IF PRINUM < LOW_KEY THEN
0870
0871                            LOW_KEY     := PRINUM;
0872
0873                        IF PRINUM > HIGH_KEY THEN
0874
0875                            HIGH_KEY    := PRINUM;
0876
0877                    END;
0878
0879                    IF (
0880                    (OBJECT_TYPE = PRI)
0881                    AND
0882                    (PRIMARY = AREA)
0883                    ) THEN
0884
0885                    BEGIN
0886
0887                        FOUND_AREA      := TRUE;
0888
0889                        IF PRINUM < LOW_AREA THEN
```

```
0890
0891                        LOW_AREA    := PRINUM;
0892
0893                    IF PRINUM > HIGH_AREA THEN
0894
0895                        HIGH_AREA   := PRINUM;
0896
0897                END;
0898
0899            END;    { WITH DEF_CURRENT^ DO }
0900
0901            DEF_CURRENT     := DEF_CURRENT^.FORE;
0902
0903        UNTIL DEF_CURRENT = NIL;
0904
0905        IF (
0906        ((FATAL) OR (HIGH_KEY <> 0))
0907        AND
0908        (NOT FOUND_0)
0909        ) THEN
0910
0911        BEGIN
0912
0913            WRITELN (SHIFT,ANSI_REVERSE,
0914            ' There is no Primary Key in the Current Definition. ',
0915            ANSI_RESET);
0916
0917            IF AUTO_TUNE THEN
0918
0919                LIB$STOP (EDF$_INSFANL,0,0,0)
0920
0921            ELSE
0922
0923            BEGIN
0924
0925                LIB$WAIT (3.0);
0926                LIB$SIGNAL (EDF$_CTRLZ,0,0,0);
0927
0928            END;
0929
0930        END;
0931
0932    END;    { SCAN_DEFINITION }
```

```
0934            { ++
0935
0936            PARSE_INPUT -- Routine to parse input string.
0937
0938            This routine will look at the chosen LIB$TPARSE table.
0939
0940            CALLING SEQUENCE:
0941
0942            PARSE_INPUT (KEY_TABLE_PTR,STATE_TABLE_PTR,DEFAULT_OK,DEFAULT_VALUE);
0943
0944            INPUT PARAMETERS:
0945
0946            KEY_TABLE_PTR
0947            STATE_TABLE_PTR
0948
0949            IMPLICIT INPUTS:
0950
0951            none
0952
0953            OUTPUT PARAMETERS:
0954
0955            none
0956
0957            IMPLICIT OUTPUTS:
0958
0959            INPUT_VALUE
0960
0961            ROUTINES CALLED:
0962
0963            none
0964
0965            ROUTINE VALUE:
0966
0967            none
0968
0969            SIGNALS:
0970
0971            none
0972
0973            SIDE EFFECTS:
0974
0975            none
0976
0977            -- }
```

```
0979        PROCEDURE PARSE_INPUT (
0980                                KEY_TABLE       : INTEGER;
0981                                STATE_TABLE     : INTEGER;
0982                                DEFAULT_OK      : BOOLEAN;
0983                                DEFAULT_VALUE   : INTEGER
0984                                );
0985
0986        BEGIN
0987
0988            { +
0989            Get the input from the terminal.
0990            - }
0991            INPUT_DESC  := NULL_STRING;
0992
0993            { +
0994            If auto answers are enabled and this question has a default - use it.
0995            - }
0996            IF (
0997            (
0998            (TAKE_DEFAULTS)
0999            AND
1000            (IDATA[EDF$K_RESPONSES] = EDF$K_AUTO)
1001            AND
1002            (DEFAULT_OK)
1003            AND
1004            (NOT (QTAB_OFFSET = EDF$K_RETURN))
1005            )
1006            OR
1007            (AUTO_TUNE)
1008            ) THEN
1009
1010            BEGIN
1011
1012                IF NOT AUTO_TUNE THEN
1013
1014                BEGIN
1015
1016                    WRITELN (CRLF);
1017                    LIB$WAIT (0.7);
1018
1019                END;
1020
1021            END
1022
1023            ELSE
1024
1025            BEGIN
1026
1027                IF EOF (INPUT) THEN
1028
1029                BEGIN
1030
1031                    RESET (INPUT);
1032                    LIB$SIGNAL (EDF$_CTRLZ,0,0,0);
1033
1034                END;
1035
```

```
1036                    READLN (INPUT_STRING);
1037                    WRITELN (CRLF);
1038                    STR$TRIM (INPUT_DESC,INPUT_STRING);
1039                    STR$UPCASE (INPUT_DESC,INPUT_DESC);
1040
1041                END;
1042
1043                { +
1044                If we're journaling our input, save a copy of it to the
1045                journal file.
1046                - }
1047                IF JOURNAL_ENABLED THEN
1048
1049                    IF INPUT_DESC.DSC$W_LENGTH > 0 THEN
1050
1051                        WRITELN (
1052                            JOURNAL_FILE,
1053                            INPUT_DESC.DSC$A_POINTER^:INPUT_DESC.DSC$W_LENGTH
1054                            )
1055
1056                    ELSE
1057
1058                        WRITELN (JOURNAL_FILE);
1059
1060                { +
1061                See if the answer was defaulted, and if it's allowed to be.
1062                - }
1063                IF INPUT_DESC.DSC$W_LENGTH = 0 THEN
1064
1065                    IF DEFAULT_OK THEN
1066
1067                        INPUT_VALUE         := DEFAULT_VALUE
1068
1069                    ELSE
1070
1071                    BEGIN
1072
1073                        LIB$SIGNAL (EDF$_NODEFAULT,0,0,0);
1074
1075                    END
1076
1077                ELSE
1078
1079                BEGIN
1080
1081                    { +
1082                    See if it's valid and get it's value.
1083                    - }
1084                    PARAM_BLOCK.TPA$L_STRINGPTR     := INPUT_DESC.DSC$A_POINTER::UNSIGNED;
1085                    PARAM_BLOCK.TPA$L_STRINGCNT     := INPUT_DESC.DSC$W_LENGTH;
1086
1087                    ISTATUS := LIB$TPARSE (
1088                                           PARAM_BLOCK,
1089                                           STATE_TABLE,
1090                                           KEY_TABLE
1091                                           );
1092
```

```
1093                    INPUT_VALUE     := PARAM_BLOCK.TPA$L_PARAM::LONG;
1094                    INPUT_NUMBER    := PARAM_BLOCK.TPA$L_NUMBER::LONG;
1095
1096                    { +
1097                    Even Istatus (low bit clear) means failure.
1098                    - }
1099                    IF NOT ODD (ISTATUS) THEN
1100
1101                    BEGIN
1102
1103                        IF PARAM_BLOCK.TPA$V_AMBIG THEN
1104
1105                            LIB$SIGNAL (EDF$_AMBIG,0,0,0)
1106
1107                        ELSE
1108
1109                            LIB$SIGNAL (EDF$_BADSYNTAX,0,0,0);
1110
1111                    END;
1112
1113                END;    { IF NOT INPUT_DESC.DSC$W_LENGTH = 0 }
1114
1115            END;    { PARSE_INPUT }
```

```
1117            { ++
1118
1119            NUMBER_INPUT -- Routine to get a number from the input string.
1120
1121            This routine will return the integer typed.
1122
1123            CALLING SEQUENCE:
1124
1125            NUMBER_INPUT (NUM_VALUE,DEFAULT_OK,DEFAULT_VALUE);
1126
1127            INPUT PARAMETERS:
1128
1129            none
1130
1131            IMPLICIT INPUTS:
1132
1133            none
1134
1135            OUTPUT PARAMETERS:
1136
1137            NUM_VALUE
1138
1139            IMPLICIT OUTPUTS:
1140
1141            none
1142
1143            ROUTINES CALLED:
1144
1145            none
1146
1147            ROUTINE VALUE:
1148
1149            none
1150
1151            SIGNALS:
1152
1153            none
1154
1155            SIDE EFFECTS:
1156
1157            none
1158
1159            -- }
```

```
1161            PROCEDURE NUMBER_INPUT (
1162                                    VAR NUM_VALUE    : INTEGER;
1163                                    DEFAULT_OK       : BOOLEAN;
1164                                    DEFAULT_VALUE    : INTEGER
1165                                    );
1166
1167            BEGIN
1168
1169                {+
1170                Get the input from the terminal.
1171                -}
1172                INPUT_DESC  := NULL_STRING;
1173
1174                {+
1175                If auto answers are enabled and this question has a default - use it.
1176                -}
1177                IF (
1178                (
1179                (TAKE_DEFAULTS)
1180                AND
1181                (IDATA[EDF$K_RESPONSES] = EDF$K_AUTO)
1182                AND
1183                (DEFAULT_OK)
1184                )
1185                OR
1186                (AUTO_TUNE)
1187                ) THEN
1188
1189                BEGIN
1190
1191                    IF NOT AUTO_TUNE THEN
1192
1193                    BEGIN
1194
1195                        WRITELN (CRLF);
1196                        LIB$WAIT (0.7);
1197
1198                    END;
1199
1200                END
1201
1202                ELSE
1203
1204                BEGIN
1205
1206                    IF EOF (INPUT) THEN
1207
1208                    BEGIN
1209
1210                        RESET (INPUT);
1211                        LIB$SIGNAL (EDF$_CTRLZ,0,0,0);
1212
1213                    END;
1214
1215                    READLN (INPUT_STRING);
1216                    WRITELN (CRLF);
1217                    STR$TRIM (INPUT_DESC,INPUT_STRING);
```

```
1218                STR$UPCASE (INPUT_DESC,INPUT_DESC);
1219                PARAM_BLOCK.TPA$L_TOKENPTR      := INPUT_DESC.DSC$A_POINTER::UNSIGNED;
1220                PARAM_BLOCK.TPA$L_TOKENCNT      := INPUT_DESC.DSC$W_LENGTH;
1221
1222            END;
1223
1224            { +
1225            If we're journaling our input, save a copy of it to the
1226            journal file.
1227            - }
1228            IF JOURNAL_ENABLED THEN
1229
1230                IF INPUT_DESC.DSC$W_LENGTH > 0 THEN
1231
1232                    WRITELN (
1233                        JOURNAL_FILE,
1234                        INPUT_DESC.DSC$A_POINTER^:INPUT_DESC.DSC$W_LENGTH
1235                        )
1236
1237                ELSE
1238
1239                    WRITELN (JOURNAL_FILE);
1240
1241            { +
1242            See if the answer was defaulted, and if it's allowed to be.
1243            - }
1244            IF INPUT_DESC.DSC$W_LENGTH = 0 THEN
1245
1246                IF DEFAULT_OK THEN
1247
1248                    NUM_VALUE           := DEFAULT_VALUE
1249
1250                ELSE
1251
1252                BEGIN
1253
1254                    LIB$SIGNAL (EDF$_NODEFAULT,0,0,0);
1255
1256                END
1257
1258            ELSE
1259
1260            BEGIN
1261
1262                { +
1263                Convert it to an integer.
1264                - }
1265                ISTATUS := OTS$CVT_TI_L (INPUT_DESC,NUM_VALUE);
1266
1267                { +
1268                Even Istatus (low bit clear) means failure.
1269                - }
1270                IF NOT ODD (ISTATUS) THEN
1271
1272                BEGIN
1273
1274                    LIB$SIGNAL (EDF$_BADSYNTAX,0,0,0);
```

```
1275
1276                    END;
1277
1278                END;    { IF NOT INPUT_DESC.DSC$W_LENGTH = 0 }
1279
1280                INPUT_VALUE         := NUM_VALUE;
1281                INPUT_NUMBER        := NUM_VALUE;
1282
1283            END;    { NUMBER_INPUT }
```

```
1285            { ++
1286
1287            MAKE_SCRATCH -- Create a new peice of dynamic memory and init it.
1288
1289            This routine creates a new Line_object, and inits its various fields.
1290
1291            CALLING SEQUENCE:
1292
1293            MAKE_SCRATCH;
1294
1295            INPUT PARAMETERS:
1296
1297            none
1298
1299            IMPLICIT INPUTS:
1300
1301            LINE_OBJECT_TEMPLATE
1302
1303            OUTPUT PARAMETERS:
1304
1305            DEF_SCRATCH
1306
1307            IMPLICIT OUTPUTS:
1308
1309            none
1310
1311            ROUTINES CALLED:
1312
1313            none
1314
1315            ROUTINE VALUE:
1316
1317            none
1318
1319            SIGNALS:
1320
1321            none
1322
1323            SIDE EFFECTS:
1324
1325            none
1326
1327            -- }
```

```
1329            [ASYNCHRONOUS] PROCEDURE MAKE_SCRATCH;
1330
1331            BEGIN
1332
1333                { +
1334                Allocate some dynamic memory.
1335                - }
1336                NEW (DEF_SCRATCH);
1337
1338                { +
1339                Copy over the template.
1340                - }
1341                DEF_SCRATCH^        := LINE_OBJECT_TEMPLATE;
1342
1343            END;    { MAKE_SCRATCH }
```

```
1345            { ++
1346
1347            CURRENT_GT_TEST -- Compare def_current with test.
1348
1349            This function has the boolean value of the whether or not DEF_CURRENT is
1350            greater than TEST.
1351
1352            CALLING SEQUENCE:
1353
1354            test-val        := CURRENT_GT_TEST (TEST,SCOPE);
1355
1356            INPUT PARAMETERS:
1357
1358            SCOPE
1359
1360            IMPLICIT INPUTS:
1361
1362            DEF_CURRENT
1363
1364            OUTPUT PARAMETERS:
1365
1366            none
1367
1368            IMPLICIT OUTPUTS:
1369
1370            none
1371
1372            ROUTINES CALLED:
1373
1374            none
1375
1376            ROUTINE VALUE:
1377
1378            True if DEF_CURRENT > TEST, false if not.
1379
1380            SIGNALS:
1381
1382            none
1383
1384            SIDE EFFECTS:
1385
1386            none
1387
1388            -- }
```

```
1390        [ASYNCHRONOUS] FUNCTION CURRENT_GT_TEST (
1391                                TEST                : LINE_OBJECT;
1392                                EXACT_COMPARISON    : BOOLEAN
1393                                ) : BOOLEAN;
1394
1395        BEGIN
1396
1397            CURRENT_GT_TEST     := FALSE;
1398
1399            { *
1400            Just do a boolean assignment.
1401            - }
1402            IF EXACT_COMPARISON THEN
1403
1404            BEGIN
1405
1406                IF
1407                (PRI_SEQ[DEF_CURRENT^.PRIMARY] > PRI_SEQ[TEST.PRIMARY])
1408                THEN
1409
1410                    CURRENT_GT_TEST     := TRUE;
1411
1412                IF
1413                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1414                AND
1415                (DEF_CURRENT^.PRINUM > TEST.PRINUM)
1416                THEN
1417
1418                    CURRENT_GT_TEST     := TRUE;
1419
1420                IF
1421                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1422                AND
1423                (DEF_CURRENT^.PRINUM = TEST.PRINUM)
1424                AND
1425                (DEF_CURRENT^.SECNUM > TEST.SECNUM)
1426                THEN
1427
1428                    CURRENT_GT_TEST     := TRUE;
1429
1430                IF
1431                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1432                AND
1433                (DEF_CURRENT^.PRINUM = TEST.PRINUM)
1434                AND
1435                (DEF_CURRENT^.SECNUM = TEST.SECNUM)
1436                AND
1437                (DEF_CURRENT^.SECONDARY > TEST.SECONDARY)
1438                THEN
1439
1440                    CURRENT_GT_TEST     := TRUE;
1441
1442            END
1443
1444            ELSE
1445
1446            BEGIN
```

```
1447
1448                    IF(
1449                    PRI_SEQ[DEF_CURRENT^.PRIMARY] > PRI_SEQ[TEST.PRIMARY])
1450                    THEN
1451
1452                        CURRENT_GT_TEST      := TRUE;
1453
1454                    IF
1455                    (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1456                    AND
1457                    (DEF_CURRENT^.PRINUM > TEST.PRINUM)
1458                    THEN
1459
1460                        CURRENT_GT_TEST      := TRUE;
1461
1462                END;
1463
1464            END;    { CURRENT_GT_TEST }
```

```
1466            { ++
1467
1468            CURRENT_LT_TEST -- Compare def_current and test.
1469
1470            This function has the boolean value of the whether or not TEST greater than
1471            DEF_CURRENT.
1472
1473            CALLING SEQUENCE:
1474
1475            test-val        := CURRENT_LT_TEST (TEST,SCOPE);
1476
1477            INPUT PARAMETERS:
1478
1479            TEST
1480
1481            IMPLICIT INPUTS:
1482
1483            DEF_CURRENT
1484
1485            OUTPUT PARAMETERS:
1486
1487            none
1488
1489            IMPLICIT OUTPUTS:
1490
1491            none
1492
1493            ROUTINES CALLED:
1494
1495            none
1496
1497            ROUTINE VALUE:
1498
1499            True if DEF_CURRENT < TEST, false if not.
1500
1501            SIGNALS:
1502
1503            none
1504
1505            SIDE EFFECTS:
1506
1507            none
1508
1509            -- }
```

```
1511        [ASYNCHRONOUS] FUNCTION CURRENT_LT_TEST (
1512                                TEST                : LINE_OBJECT;
1513                                EXACT_COMPARISON    : BOOLEAN
1514                                ) : BOOLEAN;
1515
1516        BEGIN
1517
1518            CURRENT_LT_TEST     := FALSE;
1519
1520            { +
1521            Just do a boolean assignment.
1522            - }
1523            IF EXACT_COMPARISON THEN
1524
1525            BEGIN
1526
1527                IF(
1528                PRI_SEQ[DEF_CURRENT^.PRIMARY] < PRI_SEQ[TEST.PRIMARY])
1529                THEN
1530
1531                    CURRENT_LT_TEST     := TRUE;
1532
1533                IF
1534                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1535                AND
1536                (DEF_CURRENT^.PRINUM < TEST.PRINUM)
1537                THEN
1538
1539                    CURRENT_LT_TEST     := TRUE;
1540
1541                IF
1542                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1543                AND
1544                (DEF_CURRENT^.PRINUM = TEST.PRINUM)
1545                AND
1546                (DEF_CURRENT^.SECNUM < TEST.SECNUM)
1547                THEN
1548
1549                    CURRENT_LT_TEST     := TRUE;
1550
1551                IF
1552                (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1553                AND
1554                (DEF_CURRENT^.PRINUM = TEST.PRINUM)
1555                AND
1556                (DEF_CURRENT^.SECNUM = TEST.SECNUM)
1557                AND
1558                (DEF_CURRENT^.SECONDARY < TEST.SECONDARY)
1559                THEN
1560
1561                    CURRENT_LT_TEST     := TRUE;
1562
1563        END
1564
1565        ELSE
1566
1567        BEGIN
```

```
1568
1569                    IF(
1570                    PRI_SEQ[DEF_CURRENT^.PRIMARY] < PRI_SEQ[TEST.PRIMARY])
1571                    THEN
1572
1573                        CURRENT_LT_TEST     := TRUE;
1574
1575                    IF
1576                    (PRI_SEQ[DEF_CURRENT^.PRIMARY] = PRI_SEQ[TEST.PRIMARY])
1577                    AND
1578                    (DEF_CURRENT^.PRINUM < TEST.PRINUM)
1579                    THEN
1580
1581                        CURRENT_LT_TEST     := TRUE;
1582
1583                END;
1584
1585            END;    { CURRENT_LT_TEST }
```

```
1587            { ++
1588
1589            CURRENT_EQ_TEST -- Compare def_current and test.
1590
1591            This function has the boolean value of the whether or not TEST is the
1592            same as DEF_CURRENT.
1593
1594            CALLING SEQUENCE:
1595
1596            test-val        := CURRENT_EQ_TEST (TEST,SCOPE);
1597
1598            INPUT PARAMETERS:
1599
1600            SCOPE
1601
1602            IMPLICIT INPUTS:
1603
1604            DEF_CURRENT
1605
1606            OUTPUT PARAMETERS:
1607
1608            none
1609
1610            IMPLICIT OUTPUTS:
1611
1612            none
1613
1614            ROUTINES CALLED:
1615
1616            none
1617
1618            ROUTINE VALUE:
1619
1620            True if DEF_CURRENT = TEST, false if not.
1621
1622            SIGNALS:
1623
1624            none
1625
1626            SIDE EFFECTS:
1627
1628            none
1629
1630            -- }
```

```
1632            [ASYNCHRONOUS] FUNCTION CURRENT_EQ_TEST (
1633                                    TEST                : LINE_OBJECT;
1634                                    EXACT_COMPARISON    : BOOLEAN
1635                                    ) : BOOLEAN;
1636
1637            BEGIN
1638
1639                {+
1640                Just do a boolean assignment.
1641                -}
1642                IF EXACT_COMPARISON THEN
1643
1644                    CURRENT_EQ_TEST := (
1645
1646                    (TEST.OBJECT_TYPE = DEF_CURRENT^.OBJECT_TYPE)
1647                    AND
1648                    (TEST.PRIMARY = DEF_CURRENT^.PRIMARY)
1649                    AND
1650                    (TEST.PRINUM = DEF_CURRENT^.PRINUM)
1651                    AND
1652                    (TEST.SECONDARY = DEF_CURRENT^.SECONDARY)
1653                    AND
1654                    (TEST.SECNUM = DEF_CURRENT^.SECNUM)
1655
1656                    )
1657
1658                ELSE
1659
1660                    CURRENT_EQ_TEST := (
1661
1662                    (TEST.PRIMARY = DEF_CURRENT^.PRIMARY)
1663                    AND
1664                    (TEST.PRINUM = DEF_CURRENT^.PRINUM)
1665
1666                    );
1667
1668            END;    { CURRENT_EQ_TEST }
```

```
1670            { ++
1671
1672            INSERT_BEFORE_CURRENT -- Link the DEF_SCRATCH line_object into the list.
1673
1674            This routine adds DEF_SCRATCH into the list just before DEF_CURRENT.
1675
1676            CALLING SEQUENCE:
1677
1678            INSERT_BEFORE_CURRENT;
1679
1680            INPUT PARAMETERS:
1681
1682            none
1683
1684            IMPLICIT INPUTS:
1685
1686            DEF_SCRATCH
1687            DEF_CURRENT
1688            DEF_HEAD
1689
1690            OUTPUT PARAMETERS:
1691
1692            none
1693
1694            IMPLICIT OUTPUTS:
1695
1696            none
1697
1698            ROUTINES CALLED:
1699
1700            none
1701
1702            ROUTINE VALUE:
1703
1704            none
1705
1706            SIGNALS:
1707
1708            none
1709
1710            SIDE EFFECTS:
1711
1712            none
1713
1714            -- }
```

```
1716            [ASYNCHRONOUS] PROCEDURE INSERT_BEFORE_CURRENT;
1717
1718            BEGIN
1719
1720                { +
1721                Make it the new head, if we're adding it before the old head.
1722                - }
1723                IF DEF_CURRENT = DEF_HEAD THEN
1724
1725                    DEF_HEAD        := DEF_SCRATCH;
1726
1727                { +
1728                Update the fore and back pointers.
1729                - }
1730                DEF_PRED            := DEF_CURRENT^.BACK;
1731                DEF_SCRATCH^.FORE   := DEF_CURRENT;
1732                DEF_SCRATCH^.BACK   := DEF_PRED;
1733
1734                IF DEF_PRED <> NIL THEN
1735
1736                    DEF_PRED^.FORE  := DEF_SCRATCH;
1737
1738                DEF_CURRENT^.BACK   := DEF_SCRATCH;
1739
1740                { +
1741                Leave looking at the just inserted line_object.
1742                - }
1743                DEF_CURRENT         := DEF_SCRATCH;
1744
1745            END;    { INSERT_BEFORE_CURRENT }
```

```
1747            { ++
1748
1749            INSERT_AT_CURRENT -- Link the DEF_SCRATCH line_object into the list.
1750
1751            This routine adds DEF_SCRATCH into the list at DEF_CURRENT.
1752
1753            CALLING SEQUENCE:
1754
1755            INSERT_AT_CURRENT;
1756
1757            INPUT PARAMETERS:
1758
1759            none
1760
1761            IMPLICIT INPUTS:
1762
1763            DEF_SCRATCH
1764            DEF_CURRENT
1765            DEF_HEAD
1766
1767            OUTPUT PARAMETERS:
1768
1769            none
1770
1771            IMPLICIT OUTPUTS:
1772
1773            none
1774
1775            ROUTINES CALLED:
1776
1777            none
1778
1779            ROUTINE VALUE:
1780
1781            none
1782
1783            SIGNALS:
1784
1785            none
1786
1787            SIDE EFFECTS:
1788
1789            none
1790
1791            -- }
```

```
1793            [ASYNCHRONOUS] PROCEDURE INSERT_AT_CURRENT;
1794
1795            BEGIN
1796
1797                { +
1798                Make new head or tail, if we're replacing this one.
1799                - }
1800                IF DEF_CURRENT = DEF_HEAD THEN
1801
1802                    DEF_HEAD        := DEF_SCRATCH;
1803
1804                IF DEF_CURRENT = DEF_TAIL THEN
1805
1806                    DEF_TAIL        := DEF_SCRATCH;
1807
1808                { +
1809                Substitute the links to def_current with links to def_scratch.
1810                - }
1811                DEF_PRED            := DEF_CURRENT^.BACK;
1812                DEF_SUCC            := DEF_CURRENT^.FORE;
1813                DEF_SCRATCH^.FORE   := DEF_CURRENT^.FORE;
1814                DEF_SCRATCH^.BACK   := DEF_CURRENT^.BACK;
1815
1816                IF DEF_PRED <> NIL THEN
1817
1818                    DEF_PRED^.FORE  := DEF_SCRATCH;
1819
1820                IF DEF_SUCC <> NIL THEN
1821
1822                    DEF_SUCC^.BACK  := DEF_SCRATCH;
1823
1824                { +
1825                Get rid of the old def_current, and point def_current to the new king.
1826                - }
1827                DISPOSE (DEF_CURRENT);
1828
1829                DEF_CURRENT         := DEF_SCRATCH;
1830
1831            END;    { INSERT_AT_CURRENT }
```

```
1833            { ++
1834
1835            INSERT_AFTER_CURRENT -- Link the DEF_SCRATCH line_object into the list.
1836
1837            This routine adds DEF_SCRATCH to the list after DEF_CURRENT.
1838
1839            CALLING SEQUENCE:
1840
1841            INSERT_AFTER_CURRENT;
1842
1843            INPUT PARAMETERS:
1844
1845            none
1846
1847            IMPLICIT INPUTS:
1848
1849            DEF_CURRENT
1850            DEF_SCRATCH
1851            DEF_TAIL
1852
1853            OUTPUT PARAMETERS:
1854
1855            none
1856
1857            IMPLICIT OUTPUTS:
1858
1859            none
1860
1861            ROUTINES CALLED:
1862
1863            none
1864
1865            ROUTINE VALUE:
1866
1867            none
1868
1869            SIGNALS:
1870
1871            none
1872
1873            SIDE EFFECTS:
1874
1875             one
1876
1877             - }
```

```
1879            [ASYNCHRONOUS] PROCEDURE INSERT_AFTER_CURRENT;
1880
1881            BEGIN
1882
1883                { *
1884                Make it the new tail if we're adding it after the old tail.
1885                - }
1886                IF DEF_CURRENT = DEF_TAIL THEN
1887
1888                    DEF_TAIL        := DEF_SCRATCH;
1889
1890                { *
1891                Update the fore and back pointers.
1892                - }
1893                DEF_SUCC            := DEF_CURRENT^.FORE;
1894                DEF_SCRATCH^.FORE   := DEF_SUCC;
1895                DEF_SCRATCH^.BACK   := DEF_CURRENT;
1896                DEF_CURRENT^.FORE   := DEF_SCRATCH;
1897
1898                IF DEF_SUCC <> NIL THEN
1899
1900                    DEF_SUCC^.BACK  := DEF_SCRATCH;
1901
1902                { *
1903                Leave looking at the just inserted line_object.
1904                - }
1905                DEF_CURRENT         := DEF_SCRATCH;
1906
1907            END;    { INSERT_AFTER_CURRENT }
```

```
1909            { ++
1910
1911            INCR_CURRENT -- Bump DEF_CURRENT pointer one.
1912
1913            This routine makes DEF_CURRENT point to the next line_object in the list,
1914            as long as it won't fall off the end.
1915
1916            CALLING SEQUENCE:
1917
1918            INCR_CURRENT;
1919
1920            INPUT PARAMETERS:
1921
1922            none
1923
1924            IMPLICIT INPUTS:
1925
1926            DEF_CURRENT
1927
1928            OUTPUT PARAMETERS:
1929
1930            none
1931
1932            IMPLICIT OUTPUTS:
1933
1934            none
1935
1936            ROUTINES CALLED:
1937
1938            LIB$SIGNAL
1939
1940            ROUTINE VALUE:
1941
1942            none
1943
1944            SIGNALS:
1945
1946
1947            SIDE EFFECTS:
1948
1949            none
1950
1951            -- }
```

```
1953            [ASYNCHRONOUS] PROCEDURE INCR_CURRENT;
1954
1955            BEGIN
1956
1957                { +
1958                .FORE points to the next line_object in the list.
1959                - }
1960                IF DEF_CURRENT <> NIL THEN
1961
1962                    DEF_CURRENT     := DEF_CURRENT^.FORE;
1963
1964            END;    { INCR_CURRENT }
```

```
1966            { ++
1967
1968            DECR_CURRENT -- Bump back the DEF_CURRENT pointer one.
1969
1970            This routine points DEF_CURRENT to the previous line_object in the list,
1971            as long as it won't run off the end.
1972
1973            CALLING SEQUENCE:
1974
1975            DECR_CURRENT;
1976
1977            INPUT PARAMETERS:
1978
1979            none
1980
1981            IMPLICIT INPUTS:
1982
1983            DEF_CURRENT
1984
1985            OUTPUT PARAMETERS:
1986
1987            none
1988
1989            IMPLICIT OUTPUTS:
1990
1991            none
1992
1993            ROUTINES CALLED:
1994
1995            LIB$SIGNAL
1996
1997            ROUTINE VALUE:
1998
1999            none
2000
2001            SIGNALS:
2002
2003
2004            SIDE EFFECTS:
2005
2006            none
2007
2008            -- }
```

```
2010            [ASYNCHRONOUS] PROCEDURE DECR_CURRENT;
2011
2012            BEGIN
2013
2014                { +
2015                .BACK points to the previous line_object in the list.
2016                - }
2017                IF DEF_CURRENT <> NIL THEN
2018
2019                    DEF_CURRENT     := DEF_CURRENT^.BACK;
2020
2021            END;    { DECR_CURRENT }
```

```
2023          { ++
2024
2025          NEW_IDENT_LINE -- Make a new Ident and stuff it into the definition.
2026
2027          This routine gets the date and time from the system and makes a new Ident
2028          Line_object, and puts it into the definition linked list.
2029
2030          IT ASSUMES THAT THE LIST IS COMPLETELY EMPTY!!!
2031
2032          CALLING SEQUENCE:
2033
2034          NEW_IDENT_LINE;
2035
2036          INPUT PARAMETERS:
2037
2038          none
2039
2040          IMPLICIT INPUTS:
2041
2042          DEF_SCRATCH
2043          IDENT_STRING
2044
2045          OUTPUT PARAMETERS:
2046
2047          none
2048
2049          IMPLICIT OUTPUTS:
2050
2051          DEF_SCRATCH
2052          DEF_HEAD
2053          DEF_TAIL
2054          DEF_CURRENT
2055
2056          ROUTINES CALLED:
2057
2058          LIB$DATE_TIME
2059          MAKE_SCRATCH
2060
2061          ROUTINE VALUE:
2062
2063          none
2064
2065          SIGNALS:
2066
2067          none
2068
2069          SIDE EFFECTS:
2070
2071          none
2072
2073          -- }
```

```
2075            PROCEDURE NEW_IDENT_LINE;
2076
2077            VAR
2078                DATE_STRING : STRING20;
2079                I           : INTEGER;
2080
2081            BEGIN
2082
2083                { +
2084                Create a place for the Ident to go.
2085                - }
2086                MAKE_SCRATCH;
2087
2088                { +
2089                Get system date and time to put into IDENT line.
2090                - }
2091                LIB$DATE_TIME (DATE_STRING);
2092
2093                { +
2094                Now, copy it into the ident string.
2095                - }
2096                FOR I := 1 TO 20 DO
2097
2098                    IDENT_STRING[I]         := DATE_STRING[I];
2099
2100                { +
2101                Put an IDENT primary at the head of the linked list.
2102                - }
2103                WITH DEF_SCRATCH^ DO
2104
2105                BEGIN
2106
2107                    TEMP_DESCRIPTOR                 := NULL_STRING;
2108                    NEW (TEMP_DESCRIPTOR.DSC$A_POINTER);
2109
2110                    TEMP_DESCRIPTOR.DSC$W_LENGTH    := IDENT_STRING_LENGTH;
2111                    OBJECT_TYPE                     := PRI;
2112                    PRIMARY                         := IDENT;
2113
2114                    FOR I := 1 TO IDENT_STRING_LENGTH DO
2115
2116                        TEMP_DESCRIPTOR.DSC$A_POINTER^[I]   := IDENT_STRING[I];
2117
2118                    LIB$SCOPY_DXDX (TEMP_DESCRIPTOR,STRING);
2119                    DISPOSE (TEMP_DESCRIPTOR.DSC$A_POINTER);
2120
2121                END;        { WITH DEF_SCRATCH^ }
2122
2123                { +
2124                Make the just created line_object the head (and only) one
2125                - }
2126                DEF_CURRENT         := DEF_SCRATCH;
2127                DEF_HEAD            := DEF_SCRATCH;
2128                DEF_TAIL            := DEF_SCRATCH;
2129
2130            END;    { NEW_IDENT_LINE }
```

```
2132            { ++
2133
2134            DELETE_CURRENT -- Unlink DEF_CURRENT from the list and kill it.
2135
2136            This routine removes the line_object pointed to by DEF_CURRENT from the list.
2137
2138            CALLING SEQUENCE:
2139
2140            DELETE_CURRENT;
2141
2142            INPUT PARAMETERS:
2143
2144            none
2145
2146            IMPLICIT INPUTS:
2147
2148            DEF_CURRENT
2149            DEF_TAIL
2150            DEF_HEAD
2151
2152            OUTPUT PARAMETERS:
2153
2154            none
2155
2156            IMPLICIT OUTPUTS:
2157
2158            DEF_CURRENT
2159            DEF_TAIL
2160            DEF_HEAD
2161
2162            ROUTINES CALLED:
2163
2164            EXTRACT_CURRENT
2165
2166            ROUTINE VALUE:
2167
2168            none
2169
2170            SIGNALS:
2171
2172            none
2173
2174            SIDE EFFECTS:
2175
2176            none
2177
2178            -- }
```

```
2180        PROCEDURE DELETE_CURRENT;
2181
2182        BEGIN
2183
2184            IF DEF_CURRENT^.PRIMARY = TITLE THEN
2185
2186            BEGIN
2187
2188                { +
2189                TITLE is always the very 1st line_object in the list.
2190                - }
2191                IF DEF_CURRENT^.FORE = NIL THEN
2192
2193                BEGIN
2194
2195                    DISPOSE (DEF_CURRENT);
2196                    NEW_IDENT_LINE;
2197
2198                END     { IF TRUE DEF_CURRENT^.FORE = NIL }
2199
2200                ELSE
2201
2202                BEGIN
2203
2204                    DEF_HEAD            := DEF_CURRENT^.FORE;
2205                    DEF_HEAD^.BACK      := NIL;
2206                    DISPOSE (DEF_CURRENT);
2207                    DEF_CURRENT         := DEF_HEAD;
2208
2209                END;    { IF FALSE DEF_CURRENT^.FORE = NIL }
2210
2211            END         { IF TRUE DEF_CURRENT^.PRIMARY = TITLE }
2212
2213            ELSE
2214
2215            BEGIN
2216
2217            { +
2218            Make new tail, if we're deleting old tail.
2219            - }
2220            IF (DEF_CURRENT <> NIL) AND (DEF_CURRENT = DEF_TAIL) THEN
2221
2222                DEF_TAIL        := DEF_CURRENT^.BACK;
2223
2224            { +
2225            Make new head, if we're deleting old head.
2226            - }
2227            IF (DEF_CURRENT <> NIL) AND (DEF_CURRENT = DEF_HEAD) THEN
2228
2229                DEF_HEAD        := DEF_CURRENT^.FORE;
2230
2231            { +
2232            Update fore and back pointers.
2233            - }
2234            DEF_PRED            := DEF_CURRENT^.BACK;
2235            DEF_SUCC            := DEF_CURRENT^.FORE;
2236
```

```
2237            IF DEF_PRED <> NIL THEN
2238
2239                DEF_PRED^.FORE  := DEF_SUCC;
2240
2241            IF DEF_SUCC <> NIL THEN
2242
2243                DEF_SUCC^.BACK  := DEF_PRED;
2244
2245            WITH DEF_CURRENT^ DO
2246
2247            BEGIN
2248
2249                IF STRING.DSC$W_LENGTH > 0 THEN
2250
2251                    STR$FREE1_DX (STRING);
2252
2253                IF COMMENT.DSC$W_LENGTH > 0 THEN
2254
2255                    STR$FREE1_DX (COMMENT);
2256
2257            END;
2258
2259            DISPOSE (DEF_CURRENT);
2260
2261            IF DEF_SUCC <> NIL THEN
2262
2263                DEF_CURRENT     := DEF_SUCC
2264
2265            ELSE IF DEF_PRED <> NIL THEN
2266
2267                DEF_CURRENT     := DEF_PRED
2268
2269            ELSE
2270
2271                NEW_IDENT_LINE;
2272
2273            END;        { IF FALSE DEF_CURRENT^.PRIMARY = TITLE }
2274
2275        END;    { DELETE_CURRENT }
```

```
2277            { ++
2278
2279            DELETE_PRIMARY_SECTION -- Get rid of a whole primary section.
2280
2281            This routine removes all the line_objects of particular primary from the
2282            definition linked list.
2283
2284            CALLING SEQUENCE:
2285
2286            DELETE_PRIMARY_SECTION (PRIMARY,PRINUM);
2287
2288            INPUT PARAMETERS:
2289
2290            PRIMARY
2291            PRINUM
2292
2293            IMPLICIT INPUTS:
2294
2295            DEF_CURRENT
2296            DEF_TAIL
2297            DEF_HEAD
2298
2299            OUTPUT PARAMETERS:
2300
2301            none
2302
2303            IMPLICIT OUTPUTS:
2304
2305            DEF_CURRENT
2306            DEF_TAIL
2307            DEF_HEAD
2308
2309            ROUTINES CALLED:
2310
2311            EXTRACT_CURRENT
2312
2313            ROUTINE VALUE:
2314
2315            none
2316
2317            SIGNALS:
2318
2319            none
2320
2321            SIDE EFFECTS:
2322
2323            none
2324
2325            -- }
```

```
2327        PROCEDURE DELETE_PRIMARY_SECTION (WHICHPRIMARY : PRIMARY_TYPE;
2328                                          WHICHPRINUM : INTEGER);
2329
2330        VAR
2331            DOING       : BOOLEAN;
2332            DONE        : BOOLEAN;
2333
2334        BEGIN
2335
2336            DEF_CURRENT := DEF_HEAD;
2337            DOING       := FALSE;
2338            DONE        := FALSE;
2339
2340            { +
2341            Cycle until we've deleted the section or we're at the end of the list.
2342            - }
2343            REPEAT
2344
2345                { +
2346                If this is the start of the right primary, flag it.
2347                - }
2348                IF (
2349                (DEF_CURRENT^.OBJECT_TYPE = PRI)
2350                AND
2351                (DEF_CURRENT^.PRIMARY = WHICHPRIMARY)
2352                AND
2353                (DEF_CURRENT^.PRINUM = WHICHPRINUM)
2354                ) THEN
2355
2356                    DOING       := TRUE;
2357
2358                { +
2359                If we're in the right primary, delete the sucker!
2360                - }
2361                IF DOING THEN
2362
2363                    DELETE_CURRENT
2364
2365                ELSE
2366
2367                    { +
2368                    Move on to the next line_object in the list.
2369                    - }
2370                    INCR_CURRENT;
2371
2372                { +
2373                If we're not already off the end, see if this is still the
2374                right primary. If not, flag that we're done.
2375                - }
2376                IF DEF_CURRENT <> NIL THEN
2377
2378                    IF (
2379                    (DOING)
2380                    AND
2381                    (DEF_CURRENT^.OBJECT_TYPE = PRI)
2382                    AND
2383                        (
```

```
2384                        (DEF_CURRENT^.PRIMARY <> WHICHPRIMARY)
2385                        OR
2386                        (DEF_CURRENT^.PRINUM <> WHICHPRINUM)
2387                        )
2388                    ) THEN
2389
2390                        DONE     := TRUE;
2391
2392            UNTIL (DONE) OR (DEF_CURRENT = NIL);
2393
2394        END;    { DELETE_PRIMARY_SECTION }
```

```
2396          { ++
2397
2398          INIT_DEF -- Clear out the definition and setup for a new one.
2399
2400          This routine makes room to put a brand new definition in the linked list.
2401
2402          CALLING SEQUENCE:
2403
2404          INIT_DEF;
2405
2406          INPUT PARAMETERS:
2407
2408          none
2409
2410          IMPLICIT INPUTS:
2411
2412          none
2413
2414          OUTPUT PARAMETERS:
2415
2416          none
2417
2418          IMPLICIT OUTPUTS:
2419
2420          DEF_CURRENT
2421          DEF_HEAD
2422
2423          ROUTINES CALLED:
2424
2425          none
2426
2427          ROUTINE VALUE:
2428
2429          none
2430
2431          SIGNALS:
2432
2433          none
2434
2435          SIDE EFFECTS:
2436
2437          none
2438
2439          -- }
```

```
2441            PROCEDURE INIT_DEF;
2442
2443            BEGIN
2444
2445                { *
2446                Clear out the list starting at the beginning (if not already empty).
2447                - }
2448                DEF_CURRENT := DEF_HEAD;
2449
2450                IF DEF_CURRENT <> NIL THEN
2451
2452                BEGIN
2453
2454                    REPEAT
2455
2456                        DELETE_CURRENT;
2457
2458                    UNTIL DEF_HEAD = DEF_TAIL;
2459
2460                    IF DEF_CURRENT <> NIL THEN
2461
2462                        DELETE_CURRENT;
2463
2464                END;
2465
2466            END;    { INIT_DEF }
```

```
2468            { ++
2469
2470            INSERT_IN_ORDER -- Put DEF_SCRATCH into the list in proper order.
2471
2472            This routine places the line_object pointed to by def_scratch in the definition
2473            linked list in its proper place.
2474
2475            CALLING SEQUENCE:
2476
2477            INSERT_IN_ORDER (COLLISION_ACTION);
2478
2479            INPUT PARAMETERS:
2480
2481            COLLISION_ACTION
2482
2483            IMPLICIT INPUTS:
2484
2485            DEF_CURRENT
2486            DEF_TAIL
2487            DEF_HEAD
2488            DEF_SCRATCH
2489
2490            OUTPUT PARAMETERS:
2491
2492            none
2493
2494            IMPLICIT OUTPUTS:
2495
2496            none
2497
2498            ROUTINES CALLED:
2499
2500            none
2501
2502            ROUTINE VALUE:
2503
2504            none
2505
2506            SIGNALS:
2507
2508            none
2509
2510            SIDE EFFECTS:
2511
2512            none
2513
2514            -- }
```

```
2516            [ASYNCHRONOUS] PROCEDURE INSERT_IN_ORDER (
2517                                       COLLISION_ACTION    : INTEGER
2518                                       );
2519
2520            VAR
2521                BACKUP_WORKED       : BOOLEAN;
2522
2523            BEGIN
2524
2525                { +
2526                1st, a little conditioning.
2527                - }
2528                IF (
2529                (DEF_SCRATCH^.OBJECT_TYPE = PRI)
2530                AND
2531                (DEF_SCRATCH^.PRIMARY <> TITLE)
2532                ) THEN
2533
2534                    DEF_SCRATCH^.STRING.DSC$W_LENGTH        := 0;
2535
2536                DEF_SCRATCH^.FORE   := NIL;
2537                DEF_SCRATCH^.BACK   := NIL;
2538
2539                { +
2540                Now, find the proper place. Start looking at the previous line_object.
2541                - }
2542                BACKUP_WORKED       := FALSE;
2543
2544                IF DEF_CURRENT <> NIL THEN
2545
2546                    IF DEF_CURRENT^.BACK <> NIL THEN
2547
2548                    BEGIN
2549
2550                        DECR_CURRENT;
2551
2552                        WHILE NOT (
2553                        (CURRENT_GT_TEST(DEF_SCRATCH^,TRUE))
2554                        OR
2555                        (CURRENT_EQ_TEST(DEF_SCRATCH^,TRUE))
2556                        OR
2557                        (DEF_CURRENT^.FORE = NIL)
2558                        ) DO
2559
2560                            INCR_CURRENT;
2561
2562                        BACKUP_WORKED       := (
2563                                               (
2564                                                   (CURRENT_LT_TEST(DEF_SCRATCH^,TRUE))
2565                                                   AND
2566                                                   (DEF_CURRENT^.FORE = NIL)
2567                                               )
2568                                               OR
2569                                                   (CURRENT_EQ_TEST(DEF_SCRATCH^,TRUE))
2570                                               );
2571
2572            END;    { IF DEF_CURRENT^.BACK <> NIL }
```

```
2573
2574            IF NOT BACKUP_WORKED THEN
2575
2576            BEGIN
2577
2578                { *
2579                The quick look didn't work, now scan the entire list.
2580                - }
2581                DEF_CURRENT     := DEF_HEAD;
2582
2583                WHILE NOT (
2584                (CURRENT_GT_TEST(DEF_SCRATCH^,TRUE))
2585                OR
2586                (CURRENT_EQ_TEST(DEF_SCRATCH^,TRUE))
2587                OR
2588                (DEF_CURRENT^.FORE = NIL)
2589                ) DO
2590
2591                    INCR_CURRENT;
2592
2593            END;        { IF NOT BACKUP_WORKED }
2594
2595            { *
2596            Now insert it according to how it was (found).
2597            - }
2598            IF CURRENT_GT_TEST(DEF_SCRATCH^,TRUE) THEN
2599
2600                INSERT_BEFORE_CURRENT
2601
2602            ELSE IF CURRENT_EQ_TEST(DEF_SCRATCH^,TRUE) THEN
2603
2604            BEGIN
2605
2606                IF COLLISION_ACTION = REPLACE_OBJ THEN
2607
2608                    INSERT_AT_CURRENT
2609
2610                ELSE IF COLLISION_ACTION = AFTER_OBJ THEN
2611
2612                    INSERT_AFTER_CURRENT;
2613
2614                { IF COLLISION_ACTION = IGNORE_OBJ THEN 'NULL-STATEMENT' }
2615
2616            END
2617
2618            ELSE IF DEF_CURRENT^.FORE = NIL THEN
2619
2620            BEGIN
2621
2622                DEF_TAIL        := DEF_CURRENT;
2623                INSERT_AFTER_CURRENT;
2624
2625            END;
2626
2627        END;    { INSERT_IN_ORDER }
```

```
2629            { ++
2630
2631            FIND_OBJECT -- Locate a line_object in the definition list.
2632
2633            This function returns with DEF_CURRENT pointing to the desired line
2634            object - if it finds it, in which case it's function value is true.
2635
2636            CALLING SEQUENCE:
2637
2638            BOOLEAN_VAR := FIND_OBJECT (LINE_OBJECT_TYPE,PRIMARY,PRINUM,SECONDARY,SECNUM);
2639
2640            INPUT PARAMETERS:
2641
2642            OBJECT_TYPE
2643            PRIMARY
2644            PRINUM
2645            SECONDARY
2646            SECNUM
2647
2648            IMPLICIT INPUTS:
2649
2650            none
2651
2652            OUTPUT PARAMETERS:
2653
2654            none
2655
2656            IMPLICIT OUTPUTS:
2657
2658            DEF_CURRENT
2659
2660            ROUTINES CALLED:
2661
2662            none
2663
2664            ROUTINE VALUE:
2665
2666            TRUE/FALSE DEPENDING ON FOUND STATUS
2667
2668            SIGNALS:
2669
2670            none
2671
2672            SIDE EFFECTS:
2673
2674            none
2675
2676            -- }
```

```
2678            FUNCTION FIND_OBJECT (
2679                                    OBJ_TYP         : LINE_OBJECT_TYPE;
2680                                    PRIM            : PRIMARY_TYPE;
2681                                    PRIMNUM         : INTEGER;
2682                                    SECO            : SECONDARY_TYPE;
2683                                    SECONUM         : INTEGER
2684                                    ) : BOOLEAN;
2685
2686            VAR
2687                TEST        : LINE_OBJECT;
2688                FOUND_IT    : BOOLEAN;
2689                PAST_IT     : BOOLEAN;
2690
2691            BEGIN
2692
2693                { *
2694                Stuff test object for comparison routine.
2695                - }
2696                TEST.OBJECT_TYPE    := OBJ_TYP;
2697                TEST.PRIMARY        := PRIM;
2698                TEST.PRINUM         := PRIMNUM;
2699                TEST.SECONDARY      := SECO;
2700                TEST.SECNUM         := SECONUM;
2701
2702                { *
2703                Start looking at head of list.
2704                - }
2705                DEF_CURRENT         := DEF_HEAD;
2706                FOUND_IT            := FALSE;
2707                PAST_IT             := FALSE;
2708
2709                IF DEF_CURRENT <> NIL THEN
2710
2711                BEGIN
2712
2713                    REPEAT
2714
2715                        FOUND_IT    := CURRENT_EQ_TEST (TEST,TRUE);
2716                        PAST_IT     := CURRENT_GT_TEST (TEST,TRUE);
2717
2718                        IF NOT FOUND_IT THEN
2719
2720                            INCR_CURRENT;
2721
2722                    UNTIL (FOUND_IT) OR (PAST_IT) OR (DEF_CURRENT = NIL);
2723
2724                END;
2725
2726                { *
2727                Function value indicates whether we found it or not.
2728                - }
2729                FIND_OBJECT                 := FOUND_IT;
2730
2731            END;    { FIND_OBJECT }
```

```
2733            { ++
2734
2735            POINT_AT_DEFINITION -- Setup the list pointers.
2736
2737            This routine makes the list pointers point at the Definition Linked List.
2738
2739            CALLING SEQUENCE:
2740
2741            POINT_AT_DEFINITION;
2742
2743            INPUT PARAMETERS:
2744
2745            none
2746
2747            IMPLICIT INPUTS:
2748
2749            none
2750
2751            OUTPUT PARAMETERS:
2752
2753            none
2754
2755            IMPLICIT OUTPUTS:
2756
2757            DEF_HEAD
2758            DEF_TAIL
2759            DEF_ANL_HEAD
2760            DEF_ANL_TAIL
2761            POINTING_DIRECTION
2762
2763            ROUTINES CALLED:
2764
2765            none
2766
2767            ROUTINE VALUE:
2768
2769            none
2770
2771            SIGNALS:
2772
2773            none
2774
2775            SIDE EFFECTS:
2776
2777            the current list is the definition list
2778
2779            -- }
```

```
2781        [GLOBAL] PROCEDURE POINT_AT_DEFINITION;
2782
2783        BEGIN
2784
2785            IF NOT POINTING_AT_DEFINITION THEN
2786
2787            BEGIN
2788
2789                DEF_ANL_HEAD            := DEF_HEAD;
2790                DEF_ANL_TAIL            := DEF_TAIL;
2791                DEF_HEAD                := DEF_SAVE_HEAD;
2792                DEF_TAIL                := DEF_SAVE_TAIL;
2793
2794                POINTING_AT_DEFINITION  := TRUE;
2795
2796            END;
2797
2798        END;    { POINT_AT_DEFINITION }
```

```
2800            { ++
2801
2802            POINT_AT_ANALYSIS -- Setup the list pointers.
2803
2804            This routine makes the list pointers point at the Analysis Linked List.
2805
2806            CALLING SEQUENCE:
2807
2808            POINT_AT_ANALYSIS;
2809
2810            INPUT PARAMETERS:
2811
2812            none
2813
2814            IMPLICIT INPUTS:
2815
2816            none
2817
2818            OUTPUT PARAMETERS:
2819
2820            none
2821
2822            IMPLICIT OUTPUTS:
2823
2824            DEF_HEAD
2825            DEF_TAIL
2826            DEF_ANL_HEAD
2827            DEF_ANL_TAIL
2828            POINTING_DIRECTION
2829
2830            ROUTINES CALLED:
2831
2832            none
2833
2834            ROUTINE VALUE:
2835
2836            none
2837
2838            SIGNALS:
2839
2840            none
2841
2842            SIDE EFFECTS:
2843
2844            the current list is the analysis list
2845
2846            -- }
```

```
2848            PROCEDURE POINT_AT_ANALYSIS;
2849
2850            BEGIN
2851
2852                IF POINTING_AT_DEFINITION THEN
2853
2854                BEGIN
2855
2856                    DEF_SAVE_HEAD           := DEF_HEAD;
2857                    DEF_SAVE_TAIL           := DEF_TAIL;
2858                    DEF_HEAD                := DEF_ANL_HEAD;
2859                    DEF_TAIL                := DEF_ANL_TAIL;
2860
2861                    POINTING_AT_DEFINITION  := FALSE;
2862
2863                END;
2864
2865            END;    { POINT_AT_ANALYSIS }
```

```
2867            { ++
2868
2869            EDF$LINE_PARSED -- Action routine for FDL$PARSE routine.
2870
2871            This routine stores into the definition database the values from FDL$PARSE.
2872
2873            CALLING SEQUENCE:
2874
2875            Called from the FDL$PARSE routine.
2876
2877            INPUT PARAMETERS:
2878
2879            none
2880
2881            IMPLICIT INPUTS:
2882
2883            FDL_BLOCK
2884
2885            OUTPUT PARAMETERS:
2886
2887            none
2888
2889            IMPLICIT OUTPUTS:
2890
2891            DEF_CURRENT
2892            DEF_TAIL
2893            DEF_HEAD
2894            DEF_SCRATCH
2895
2896            ROUTINES CALLED:
2897
2898            MAKE_SCRATCH
2899            LIB$SCOPY_DXDX
2900
2901            ROUTINE VALUE:
2902
2903            Always 1 (for success).
2904
2905            SIGNALS:
2906
2907            none
2908
2909            SIDE EFFECTS:
2910
2911            none
2912
2913            -- }
```

```
2915        [ASYNCHRONOUS,GLOBAL] FUNCTION EDF$LINE_PARSED : INTEGER;
2916
2917        BEGIN
2918
2919            { +
2920            This routine always succeeds.
2921            - }
2922            EDF$LINE_PARSED     := 1;
2923
2924            { +
2925            Create a new line object to be added to the list.
2926            - }
2927            MAKE_SCRATCH;
2928
2929            { +
2930            Get the control longword.
2931            - }
2932            TEMP_FDL3$TYPE      := FDL_BLOCK^[FDL$L_CTRL]::FDL3$TYPE;
2933
2934            { +
2935            Completely ignore a line if it's an IDENT, or if the Warning bit is set.
2936            - }
2937            IF ( NOT (
2938            ((TEMP_FDL3$TYPE.FDL$V_NEWPRI)
2939            AND
2940            (FDL_BLOCK^[FDL$L_PRIMARY]::PRIMARY_TYPE = IDENT))
2941            OR
2942            (TEMP_FDL3$TYPE.FDL$V_WARNING)
2943            )) THEN
2944
2945            WITH DEF_SCRATCH^ DO
2946
2947            BEGIN
2948
2949                { +
2950                Set the type of this line_object.
2951                - }
2952                IF TEMP_FDL3$TYPE.FDL$V_NEWPRI THEN
2953
2954                    OBJECT_TYPE := PRI
2955
2956                ELSE
2957
2958                    OBJECT_TYPE := SEC;
2959
2960                { +
2961                Check for a full line comment, as it is a 3rd object type.
2962                - }
2963                IF TEMP_FDL3$TYPE.FDL$V_LINECMT THEN
2964
2965                    OBJECT_TYPE := COM;
2966
2967                { +
2968                Fetch the primary and secondary values.
2969                - }
2970                PRIMARY         := FDL_BLOCK^[FDL$L_PRIMARY]::PRIMARY_TYPE;
2971                SECONDARY       := FDL_BLOCK^[FDL$L_SECONDARY]::SECONDARY_TYPE;
```

```
2972
2973            { +
2974            Store the comment string if a comment was detected.
2975            - }
2976            IF TEMP_FDL3$TYPE.FDL$V_COMMENT OR TEMP_FDL3$TYPE.FDL$V_LINECMT THEN
2977
2978            BEGIN
2979
2980                TEMP_DESCRIPTOR     := NULL_STRING;
2981                TEMP_DESCRIPTOR     := CVT_QUAD_DESC (
2982                                                      FDL_BLOCK^[FDL$Q_COMMENT],
2983                                                      FDL_BLOCK^[FDL$Q_COMMENT+1]
2984                                                      );
2985                LIB$SCOPY_DXDX (TEMP_DESCRIPTOR,COMMENT);
2986
2987            END;
2988
2989            { +
2990            Store the string if it's an attribute with a string value.
2991            - }
2992            IF (
2993
2994            ((NOT TEMP_FDL3$TYPE.FDL$V_NEWPRI) AND (SEC_TYPE[SECONDARY].STR))
2995
2996            OR
2997                    { for positioning by file name }
2998            ((NOT TEMP_FDL3$TYPE.FDL$V_NEWPRI) AND (SECONDARY = POSITION$))
2999
3000            OR
3001
3002            ((TEMP_FDL3$TYPE.FDL$V_NEWPRI) AND (PRIMARY = TITLE))
3003                                                                                    ; F
3004            ) THEN
3005
3006            BEGIN
3007
3008                TEMP_DESCRIPTOR     := NULL_STRING;
3009                TEMP_DESCRIPTOR     := CVT_QUAD_DESC (
3010                                                      FDL_BLOCK^[FDL$Q_STRING],
3011                                                      FDL_BLOCK^[FDL$Q_STRING+1]
3012                                                      );
3013                LIB$SCOPY_DXDX (TEMP_DESCRIPTOR,STRING);
3014
3015            END;
3016
3017            { +
3018            Now stuff the new line_object with the remaining data from FDL$PARSE.
3019            - }
3020            IF PRIMARY IN [ AREA, KEY, ANALYSIS_OF_AREA, ANALYSIS_OF_KEY ] THEN
3021
3022                PRINUM      := FDL_BLOCK^[FDL$L_PRINUM]
3023
3024            ELSE
3025
3026                PRINUM      := 0;
3027
3028            IF SECONDARY IN [ SEG_LENGTH, SEG_POSITION ] THEN
```

```
3029
3030                        SECNUM      := FDL_BLOCK^[FDL$L_SECNUM]
3031
3032                    { +
3033                    Until RMS supports different types per segment,
3034                    make the Type secondary that last one in the Key primary.
3035                    (SECNUM is higher sorting priority than SECONDARY)
3036                    - }
3037                    ELSE IF SECONDARY = SEG_TYPE THEN
3038
3039                        SECNUM      := 7
3040
3041                    ELSE
3042
3043                        SECNUM      := 0;
3044
3045                { ++
3046
3047                IF RMS EVER IMPLEMENTS DIFFERENT DATA TYPES FOR EACH KEY SEGMENT,
3048                USE THE FOLLOWING CODE.
3049
3050                    IF SECONDARY IN [ SEG_LENGTH, SEG_POSITION, SEG_TYPE ] THEN
3051
3052                        SECNUM      := FDL_BLOCK^[FDL$L_SECNUM]
3053
3054                    ELSE
3055
3056                        SECNUM      := 0;
3057
3058                -- }
```

```
3060                    { +
3061                    Qualifiers values are more complicated.
3062                    - }
3063                    IF (
3064                    (SECONDARY = MT_PROTECTION)
3065                    OR
3066                    (SECONDARY = NULL_VALUE)
3067                    ) THEN
3068
3069                    BEGIN
3070
3071                        { +
3072                        These two come back in a funny place.
3073                        - }
3074                        QUALIFIER    := 0;
3075                        NUMBER       := FDL_BLOCK^[FDL$L_QUALIFIER];
3076
3077                    END
3078
3079                    ELSE
3080
3081                    BEGIN
3082
3083                        { +
3084                        See which secondary we have.
3085                        - }
3086                        CASE SECONDARY OF
3087
3088                            ORGANIZATION :
3089
3090                                CASE FDL_BLOCK^[FDL$L_QUALIFIER] OF
3091
3092                                    FAB$C_IDX :     QUALIFIER       := FDL$C_IDX;
3093                                    FAB$C_REL :     QUALIFIER       := FDL$C_REL;
3094                                    FAB$C_SEQ :     QUALIFIER       := FDL$C_SEQ;
3095
3096                                OTHERWISE
3097
3098                                    { NULL-STATEMENT } ;
3099
3100                                END;        { CASE FDL_BLOCK^[FDL$L_QUALIFIER] }
3101
3102                            FORMAT :
3103
3104                                CASE FDL_BLOCK^[FDL$L_QUALIFIER] OF
3105
3106                                    FAB$C_FIX :     QUALIFIER       := FDL$C_FIX;
3107                                    FAB$C_STM :     QUALIFIER       := FDL$C_STM;
3108                                    FAB$C_STMCR :   QUALIFIER       := FDL$C_STMCR;
3109                                    FAB$C_STMLF :   QUALIFIER       := FDL$C_STMLF;
3110                                    FAB$C_UDF :     QUALIFIER       := FDL$C_UDF;
3111                                    FAB$C_VAR :     QUALIFIER       := FDL$C_VAR;
3112                                    FAB$C_VFC :     QUALIFIER       := FDL$C_VFC;
3113
3114                                OTHERWISE
3115
3116                                    { NULL-STATEMENT } ;
```

```
3117
3118                    END;        { CASE FDL_BLOCK^[FDL$L_QUALIFIER] }
3119
3120                SEG_TYPE :
3121
3122                    CASE FDL_BLOCK^[FDL$L_QUALIFIER] OF
3123
3124                        XAB$C_BN2 :     QUALIFIER       := FDL$C_BN2;
3125                        XAB$C_BN4 :     QUALIFIER       := FDL$C_BN4;
3126                        XAB$C_BN8 :     QUALIFIER       := FDL$C_BN8;
3127                        XAB$C_PAC :     QUALIFIER       := FDL$C_PAC;
3128                        XAB$C_IN2 :     QUALIFIER       := FDL$C_IN2;
3129                        XAB$C_IN4 :     QUALIFIER       := FDL$C_IN4;
3130                        XAB$C_IN8 :     QUALIFIER       := FDL$C_IN8;
3131                        XAB$C_STG :     QUALIFIER       := FDL$C_STG;
3132
3133                    OTHERWISE
3134
3135                        { NULL-STATEMENT } ;
3136
3137                    END;        { CASE FDL_BLOCK^[FDL$L_QUALIFIER] }
3138
3139            OTHERWISE
3140
3141                QUALIFIER       := FDL_BLOCK^[FDL$L_QUALIFIER];
3142
3143            END;        { CASE SECONDARY }
3144
3145            NUMBER      := FDL_BLOCK^[FDL$L_NUMBER];
3146
3147        END;
3148
3149        { +
3150        Now store the other information coming back from FDL$PARSE.
3151        - }
3152        IF ODD (FDL_BLOCK^[FDL$L_SWITCH]) THEN
3153
3154            SWITCH      := TRUE
3155
3156        ELSE
3157
3158            SWITCH      := FALSE;
3159
3160        OWNER_UIC       := FDL_BLOCK^[FDL$L_OWNER_UIC];
3161        PROT_MASK       := FDL_BLOCK^[FDL$L_PROTECTION]::CTRL_ARRAY;
3162        FID1            := FDL_BLOCK^[FDL$L_FID1];
3163        FID2            := FDL_BLOCK^[FDL$L_FID2];
3164        FID3            := FDL_BLOCK^[FDL$L_FID3];
3165
3166        { +
3167        Now put def_scratch into the linked list. Depending upon whether
3168        we're inputting an FDL file or a analysis file.
3169        CLUSTER_SIZE must go into both the analysis definition and the
3170        m  n definition.
3171        - }
3172        IF (
3173        (ANALYSIS_ONLY AND (PRIMARY IN [ ANALYSIS_OF_KEY, ANALYSIS_OF_AREA ]))
```

```
3174                OR
3175                ((NOT ANALYSIS_ONLY) AND (NOT (PRIMARY IN [ ANALYSIS_OF_KEY,
3176                ANALYSIS_OF_AREA ])))
3177                OR
3178                (SECONDARY IN [ CLUSTER_SIZE ])
3179                ) THEN
3180
3181                    IF OBJECT_TYPE = COM THEN
3182
3183                    BEGIN
3184
3185                        DEF_CURRENT     := DEF_TAIL;
3186                        INSERT_AFTER_CURRENT;
3187
3188                    END
3189
3190                    ELSE
3191
3192                        INSERT_IN_ORDER (IGNORE_OBJ);
3193
3194            END;        { IF NOT IDENT PRIMARY; ALSO WITH DEF_SCRATCH^ DO }
3195
3196        END;    { EDF$LINE_PARSED }
3197
3198        END.
3199            { End of file: SRC$:EDFUTIL.PAS }
```

```
                                                
```

```
                                5C       0C   BC   D0 0000A          MOVL    @12(R12),MAX
                                50            51   D1 0000E          CMPL    VALUE,BASE                                  ; 0278
                                              00V  19 00011          BLSS    2$
                                              50   D5 00013          TSTL    BASE
                                              00V  12 00015          BNEQ    3$
                                52            50   D0 00017 2$:      MOVL    BASE,TEMP                                   ; 0280
                                              00V  11 0001A          BRB     7$
                          52                  51   50   C7 0001C 3$:      DIVL3   BASE,VALUE,TEMP                     ; 0286
                    53    51                  00   00   7A 00020          EMUL    #0,#0,VALUE,R3                      ; 0288
                    53    53                  53   50   7B 00025          EDIV    BASE,R3,R3,R3
                                              53   D5 0002A          TSTL    R3
                                              00V  18 0002C          BGEQ    4$
                                53            50   C0 0002E          ADDL2   BASE,R3
                                              53   D5 00031 4$:      TSTL    R3
                                              00V  13 00033          BEQL    6$
                                              52   D6 00035          INCL    TEMP                                        ; 0290
                                52            50   C4 00037 6$:      MULL2   BASE,TEMP                                   ; 0292
                                5C            52   D1 0003A 7$:      CMPL    TEMP,MAX                                    ; 0296
                                              00V  15 0003D          BLEQ    9$
                                52            5C   D0 0003F          MOVL    MAX,TEMP                                    ; 0298
                                50            52   D0 00042 9$:      MOVL    MAX_FACTOR,R0                               ; 0302
                                                   04 00045          RET

; Routine Size: 70 bytes,    Routine Base: $CODE + 00EC

                                                        00000 CALC_REC_OVERHEAD:                                         ; 0348
                                                   0000 00000          .WORD   ^M<>
                                5C       04   BC   D0 00002          MOVL    @4(R12),INDEX_LEVEL
                                              50   D4 00006          CLRL    RECORD_OVERHEAD                             ; 0357
                                   00000084G  EF   D5 00008          TSTL    IDATA+132                                   ; 0362
                                              00V  13 0000E          BEQL    3$
                                              5C   D5 00010          TSTL    INDEX_LEVEL
                                              00V  12 00012          BNEQ    3$
                                50            0B   C0 00014          ADDL2   #11,RECORD_OVERHEAD                         ; 0364
                    00V0000000CG  EF            00   E1 00017 3$:      BBC     #0,BDATA+12,5$                          ; 0369
                                              5C   D5 0001F          TSTL    INDEX_LEVEL
                                              00V  13 00021          BEQL    7$
                    00V0000000EG  EF            00   E1 00023 5$:      BBC     #0,BDATA+14,8$
                                              5C   D5 0002B          TSTL    INDEX_LEVEL
                                              00V  13 0002D          BEQL    8$
                                50            02   C0 0002F 7$:      ADDL2   #2,RECORD_OVERHEAD                          ; 0375
                                              5C   D5 00032 8$:      TSTL    INDEX_LEVEL                                 ; 0380
                                              00V  13 00034          BEQL    10$
                                50            04   C0 00036          ADDL2   #4,RECORD_OVERHEAD                          ; 0382
                                   00000084G  EF   D5 00039 10$:     TSTL    IDATA+132                                   ; 0387
                                              00V  12 0003F          BNEQ    21$
                                              5C   D5 00041          TSTL    INDEX_LEVEL
                                              00V  12 00043          BNEQ    21$
                    00V000000000G  EF            00   E1 00045          BBC     #0,VARIABLE_RECORDS,14$                 ; 0391
                                50            0B   C0 0004D          ADDL2   #11,RECORD_OVERHEAD                         ; 0393
                                              00V  11 00050          BRB     15$
                                50            09   C0 00052 14$:     ADDL2   #9,RECORD_OVERHEAD                          ; 0397
                    00V0000000DG  EF            00   E1 00055 15$:     BBC     #0,BDATA+13,17$                         ; 0399
                                50            03   C0 0005D          ADDL2   #3,RECORD_OVERHEAD                          ; 0401
                    00V0000000CG  EF            00   E0 00060 17$:     BBS     #0,BDATA+12,19$                         ; 0403
                    00V0000000DG  EF            00   E1 00068          BBC     #0,BDATA+13,21$
                                50   000000D8G   EF   C0 00070 19$:     ADDL2   IDATA+216,RECORD_OVERHEAD                   ; 0405
```

```
                                        04 00077 21$:    RET                                                    ; 0411

; Routine Size: 120 bytes,    Routine Base: $CODE + 00132

                                        00000 CALC_BUC_OVERHEAD:                                                ; 0457
                                   0000 00000            .WORD   ^M<>
                         5C        04   BC   D0 00002            MOVL    @4(R12),INDEX_LEVEL
                                        00V  12 00006            BNEQ    2$                                     ; 0463
                         5C             10   D0 00008            MOVL    #16,CALC_BUC_OVERHEAD                  ; 0465
                                        00V  11 0000B            BRB     3$
                         5C             12   D0 0000D 2$:        MOVL    #18,CALC_BUC_OVERHEAD                  ; 0469
                         50             5C   D0 00010 3$:        MOVL    CALC_BUC_OVERHEAD,R0                   ; 0471
                                             04 00013            RET

; Routine Size: 20 bytes,    Routine Base: $CODE + 001AA

                                   0000 00000            .ENTRY  EDF$RESET_SCROLL,^M<>                          ; 0520
                         5E             08   C2 00002            SUBL2   #8,SP
                  00V00000000G    EF             00   E0 00005            BBS     #0,AUTO_TUNE,8$                ; 0524
                  00V00000000G    EF             00   E1 0000D            BBC     #0,REGIS,4$                    ; 0531
                                  00000000G      EF   D4 00015            CLRL    CHFFLAGS                       ; 0535
                                  00000000G      EF   B4 0001B            CLRW    OUT_LINE                       ; 0536
                                  FFFFFE1B       EF   9F 00021            PUSHAB  C.AAA
                                                 02   DD 00027            PUSHL   #2
                                  00000000G      EF   9F 00029            PUSHAB  OUT_LINE
                                  000000FF       8F   DD 0002F            PUSHL   #255
                     00000000G    EF                  04   FB 00035            CALLS   #4,PAS$WRITEV_STRING
                                  00000000G      CF   9F 0003C            PUSHAB  CHFFLAGS                       ; 0537
                                  00000000G      EF   9F 00042            PUSHAB  ONE
                           F8     AD   0B2500FF  8F   D0 00048            MOVL    #186974463,-8(FP)
                           FC     AD   00000000G EF   9E 00050            MOVAB   OUT_LINE,-4(FP)
                                       F8        AD   9F 00058            PUSHAB  -8(FP)
                     00000000G    EF                  03   FB 0005B            CALLS   #3,LIB$PUT_LINE
                  00V00000000G    EF                  00   E1 00062 4$:        BBC     #0,SCROLLING_SET,7$    ; 0544
                                  00000000G      EF   9F 0006A            PUSHAB  LINES_PER_PAGE                 ; 0546
                                  00000000G      EF   9F 00070            PUSHAB  LINE_ONE
                     00000000G    EF                  02   FB 00076            CALLS   #2,LIB$SET_SCROLL
                                                           0007D 7$:
                  03 00000000G    EF                  00   E0 0007D 8$:        BBS     #0,FILE_CREATED,..+3   ; 0553
                                             0000V      31   00085            BRW     12$
                                  00000000G      EF   B5 00088            TSTW    RES_OUTPUT_FILENAME_DESC
                                                 03   1A 0008E            BGTRU   .+3
                                             0000V      31   00090            BRW     12$
                     00000000G    EF                  02   D0 00093            MOVL    #2,CHFFLAGS            ; 0561
                                  00000000G      EF   B4 0009A            CLRW    OUT_LINE                       ; 0562
                                  00000000G      EF   9F 000A0            PUSHAB  CRLF
                                                 02   DD 000A6            PUSHL   #2
                                  00000000G      EF   9F 000A8            PUSHAB  OUT_LINE
                                  000000FF       8F   DD 000AE            PUSHL   #255
                     00000000G    EF                  04   FB 000B4            CALLS   #4,PAS$WRITEV_STRING
                                  7E 00000000G   EF   3C 000BB            MOVZWL  RES_OUTPUT_FILENAME_DESC,-(SP)
                                                 00   DD 000C2            PUSHL   #0
                                  50 00000004G   EF   D0 000C4            MOVL    RES_OUTPUT_FILENAME_DESC+4,R0
                                                 60   9F 000CB            PUSHAB  (R0)
                                  000000FF       8F   DD 000CD            PUSHL   #255
                                  00000000G      EF   9F 000D3            PUSHAB  OUT_LINE
                                  000000FF       8F   DD 000D9            PUSHL   #255
```

```
                                                
```

```
                    00000000G  EF   9F 0008C          PUSHAB  LINE_ONE
   00000000G  EF                02   FB 00092          CALLS   #2,LIB$SET_CURSOR
                          0000V  31   00099          BRW     27$
00V00000000G  EF                00   E1 0009C 8$:      BBC     #0,REGIS,11$                                    ; 0660
                    FFFFFC47    EF   9F 000A4          PUSHAB  C.AAE                                           ; 0664
                                2B   DD 000AA          PUSHL   #43
                    00000000G   EF   9F 000AC          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                03   FB 000B2          CALLS   #3,PAS$WRITE_STRING
                    00000000G   EF   9F 000B9          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                01   FB 000BF          CALLS   #1,PAS$WRITELN2
                    00000000G   EF   9F 000C6          PUSHAB  COL_ONE                                         ; 0667
                    00000000G   EF   9F 000CC          PUSHAB  PROMPT_LINE
   00000000G  EF                02   FB 000D2          CALLS   #2,LIB$SET_CURSOR
                          0000V  31   000D9          BRW     27$
                    00000000G   EF   9F 000DC 11$:     PUSHAB  COL_ONE                                         ; 0673
                    00000000G   EF   9F 000E2          PUSHAB  LOWER_LINE
   00000000G  EF                02   FB 000E8          CALLS   #2,LIB$ERASE_PAGE
                          0000V  31   000EF          BRW     27$
00V00000000G  EF                00   E0 000F2 14$:     BBS     #0,FULL_PROMPT,16$                              ; 0677
03 00000000G  EF                00   E0 000FA          BBS     #0,TEMP_FULL_PROMPT,.+3
                          0000V  31   00102          BRW     27$
00V00000000G  EF                00   E1 00105 16$:     BBC     #0,TEMP_FULL_PROMPT,18$                         ; 0681
                    666640A6    8F   DF 0010D          PUSHAF  #^F1.3                                          ; 0683
   00000000G  EF                01   FB 00113          CALLS   #1,LIB$WAIT
00V00000000G  EF                00   E1 0011A 18$:     BBC     #0,REGIS,20$                                    ; 0692
                    FFFFFBF5    EF   9F 00122          PUSHAB  C.AAF                                           ; 0694
                                0B   DD 00128          PUSHL   #11
                    00000000G   EF   9F 0012A          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                03   FB 00130          CALLS   #3,PAS$WRITE_STRING
                    00000000G   EF   9F 00137          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                01   FB 0013D          CALLS   #1,PAS$WRITELN2
                    00000000G   EF   9F 00144 20$:     PUSHAB  COL_ONE                                         ; 0696
                    00000000G   EF   9F 0014A          PUSHAB  LINE_ONE
   00000000G  EF                02   FB 00150          CALLS   #2,LIB$ERASE_PAGE
                                01   DD 00157          PUSHL   #1                                              ; 0697
                                20   DD 00159          PUSHL   #32
                    00000000G   EF   9F 0015B          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                03   FB 00161          CALLS   #3,PAS$WRITE_CHAR
                    00000000G   EF   9F 00168          PUSHAB  PAS$FV_OUTPUT
   00000000G  EF                01   FB 0016E          CALLS   #1,PAS$WRITELN2
                    00000000G   EF   9F 00175          PUSHAB  COL_ONE                                         ; 0698
                    00000000G   EF   9F 0017B          PUSHAB  LINE_ONE
   00000000G  EF                02   FB 00181          CALLS   #2,LIB$SET_CURSOR
                          00V   11   00188          BRB     27$
                    0000001F    8F   DF 0018A 24$:     PUSHAL  #31                                             ; 0702
   00000000G  EF                01   FB 00190          CALLS   #1,QUERY
                          00V   11   00197          BRB     27$
                                     00199 26$:
                                     00199 27$:
                                04   00199 28$:     RET                                                     ; 0712

; Routine Size: 410 bytes,    Routine Base: $CODE + 00327

                                     00000 CVT_QUAD_DESC:                                                    ; 0759
                               0000  00000          .WORD   ^M<>
                         5E          08   C2 00002          SUBL2   #8,SP
                         50    04    BC   D0 00005          MOVL    @4(R12),LONG1
```

```
                5C      08   BC   D0 00009          MOVL    @8(R12),LONG2
                    00000000G EF   94 0000D         CLRB    QUAD_DESC                                   ; 0770
00000001G       EF            50   D0 00013         MOVL    LONG1,QUAD_DESC+1                           ; 0771
00000005G       EF            5C   D0 0001A         MOVL    LONG2,QUAD_DESC+5                           ; 0772
00000000G       EF            01   90 00021         MOVB    #1,QUAD_DESC                                ; 0777
          F8    AD 00000001G  EF   7D 00028         MOVQ    QUAD_DESC+1,CVT_QUAD_DESC                   ; 0778
                50      F8    AD   7D 00030         MOVQ    CVT_QUAD_DESC,R0                            ; 0782
                                   04 00034         RET

; Routine Size: 53 bytes,    Routine Base: $CODE + 004C1

                                   00000 SCAN_DEFINITION:                                              ; 0833
                                0000 00000         .WORD   ^M<>
                5C      04   BC   90 00002         MOVB    @4(R12),FATAL
00000000G       EF 00000000G EF   D0 00006         MOVL    DEF_HEAD,DEF_CURRENT                        ; 0840
                   00000000G EF   94 00011         CLRB    FOUND_0                                     ; 0841
                   00000000G EF   94 00017         CLRB    FOUND_AREA                                  ; 0842
                   00000000G EF   94 0001D         CLRB    FOUND_KEY                                   ; 0843
                   00000000G EF   D4 00023         CLRL    LOW_AREA                                    ; 0844
                   00000000G EF   D4 00029         CLRL    HIGH_AREA                                   ; 0845
                   00000000G EF   D4 0002F         CLRL    LOW_KEY                                     ; 0846
                   00000000G EF   D4 00035         CLRL    HIGH_KEY                                    ; 0847
                50 00000000G EF   D0 0003B 1$:     MOVL    DEF_CURRENT,R0                              ; 0851
                             60   95 00042         TSTB    (R0)                                        ; 0855
                             00V  12 00044         BNEQ    11$
                0B      19   A0   91 00046         CMPB    25(R0),#11
                             00V  12 0004A         BNEQ    11$
                          1A A0   D5 0004C         TSTL    26(R0)                                      ; 0863
                             00V  12 0004F         BNEQ    6$
00000000G       EF           01   90 00051         MOVB    #1,FOUND_0                                  ; 0865
00000000G       EF           01   90 00058 6$:     MOVB    #1,FOUND_KEY                                ; 0867
00000000G       EF        1A A0   D1 0005F         CMPL    26(R0),LOW_KEY                              ; 0869
                             00V  18 00067         BGEQ    8$
00000000G       EF        1A A0   D0 00069         MOVL    26(R0),LOW_KEY                              ; 0871
00000000G       EF        1A A0   D1 00071 8$:     CMPL    26(R0),HIGH_KEY                             ; 0873
                             00V  15 00079         BLEQ    11$
00000000G       EF        1A A0   D0 0007B         MOVL    26(R0),HIGH_KEY                             ; 0875
                             60   95 00083 11$:     TSTB    (R0)                                        ; 0879
                             00V  12 00085         BNEQ    18$
                05      19   A0   91 00087         CMPB    25(R0),#5
                             00V  12 0008B         BNEQ    18$
00000000G       EF           01   90 0008D         MOVB    #1,FOUND_AREA                               ; 0887
00000000G       EF        1A A0   D1 00094         CMPL    26(R0),LOW_AREA                             ; 0889
                             00V  18 0009C         BGEQ    15$
00000000G       EF        1A A0   D0 0009E         MOVL    26(R0),LOW_AREA                             ; 0891
00000000G       EF        1A A0   D1 000A6 15$:     CMPL    26(R0),HIGH_AREA                            ; 0893
                             00V  15 000AE         BLEQ    18$
00000000G       EF        1A A0   D0 000B0         MOVL    26(R0),HIGH_AREA                            ; 0895
                50 00000000G EF   D0 000B8 18$:     MOVL    DEF_CURRENT,R0                              ; 0901
00000000G       EF        01 A0   D0 000BF         MOVL    1(R0),DEF_CURRENT
                                03 13 000C7         BEQL    .+3
                             FF6F  31 000C9         BRW     1$
                00V             5C E8 000CC         BLBS    FATAL,21$                                   ; 0905
                   00000000G    EF D5 000CF         TSTL    HIGH_KEY
                                03 12 000D5         BNEQ    .+3
                             0000V 31 000D7         BRW     26$
03 00000000G    EF              00 E1 000DA 21$:    BBC     #0,FOUND_0,.+3
```

```
                                 0000V  31 000E2          BRW     26$
                        00000000G  EF   9F 000E5          PUSHAB  SHIFT                                   ; 0913
                                   04   DD 000EB          PUSHL   #4
                        00000000G  EF   9F 000ED          PUSHAB  PAS$FV_OUTPUT
                 00000000G  EF          03   FB 000F3     CALLS   #3,PAS$WRITE_STRING
                        00000000G  EF   9F 000FA          PUSHAB  ANSI_REVERSE
                                   04   DD 00100          PUSHL   #4
                        00000000G  EF   9F 00102          PUSHAB  PAS$FV_OUTPUT
                 00000000G  EF          03   FB 00108     CALLS   #3,PAS$WRITE_STRING
                        FFFFFA45   EF   9F 0010F          PUSHAB  C_AAG
                                   34   DD 00115          PUSHL   #52
                        00000000G  EF   9F 00117          PUSHAB  PAS$FV_OUTPUT
                 00000000G  EF          03   FB 0011D     CALLS   #3,PAS$WRITE_STRING
                        00000000G  EF   9F 00124          PUSHAB  ANSI_RESET
                                   04   DD 0012A          PUSHL   #4
                        00000000G  EF   9F 0012C          PUSHAB  PAS$FV_OUTPUT
                 00000000G  EF          03   FB 00132     CALLS   #3,PAS$WRITE_STRING
                        00000000G  EF   9F 00139          PUSHAB  PAS$FV_OUTPUT
              00000000G  EF             01   FB 0013F     CALLS   #1,PAS$WRITELN2
          00V00000000G  EF              00   E1 00146     BBC     #0,AUTO_TUNE,24$                        ; 0917
                                        00   DD 0014E     PUSHL   #0                                      ; 0919
                                        00   DD 00150     PUSHL   #0
                                        00   DD 00152     PUSHL   #0
                        00B3801C        8F   DD 00154     PUSHL   #11763740
              00000000G  EF             04   FB 0015A     CALLS   #4,LIB$STOP
                                        00V  11 00161     BRB     26$
                        00004140        8F   DF 00163 24$: PUSHAF  #^F3.0                                 ; 0925
              00000000G  EF             01   FB 00169     CALLS   #1,LIB$WAIT
                                        00   DD 00170     PUSHL   #0                                      ;.0926
                                        00   DD 00172     PUSHL   #0
                                        00   DD 00174     PUSHL   #0
                        00B3804B        8F   DD 00176     PUSHL   #11763787
              00000000G  EF             04   FB 0017C     CALLS   #4,LIB$SIGNAL
                                        04 00183 26$:     RET                                             ; 0932

; Routine Size: 388 bytes,    Routine Base: $CODE + 004F6

                                        00000 PARSE_INPUT:                                               ; 0979
                                   001C 00000             .WORD   ^M<R2,R3,R4>
                        5E              08   C2 00002     SUBL2   #8,SP
                        52        04    BC   D0 00005     MOVL    @4(R12),KEY_TABLE
                        53        08    BC   D0 00009     MOVL    @8(R12),STATE_TABLE
                        54        0C    BC   90 0000D     MOVB    @12(R12),DEFAULT_OK
                        5C        10    BC   D0 00011     MOVL    @16(R12),DEFAULT_VALUE
                 00000000G  EF 00000000G  EF   7D 00015     MOVQ    NULL_STRING,INPUT_DESC                  ; 0991
          00V00000000G  EF              00   E1 00020     BBC     #0,TAKE_DEFAULTS,4$                     ; 0996
                           00001104G    EF   D5 00028     TSTL    IDATA+260
                                        00V  12 0002E     BNEQ    4$
                        00V             54   E9 00030     BLBC    DEFAULT_OK,4$
                        1F 00000000G    EF   D1 00033     CMPL    QTAB_OFFSET,#31
                                        00V  12 00038     BNEQ    5$
          00V00000000G  EF              00   E1 0003C 4$:  BBC     #0,AUTO_TUNE,8$
          03 00000000G  EF              00   E1 00044 5$:  BBC     #0,AUTO_TUNE,.+3                        ; 1012
                                    0000V  31 0004C        BRW     12$
                           00000000G    EF   9F 0004F     PUSHAB  CRLF                                    ; 1016
                                        02   DD 00055     PUSHL   #2
                           00000000G    EF   9F 00057     PUSHAB  PAS$FV_OUTPUT
```

```
                00000000G  EF              03   FB 0005D          CALLS   #3,PAS$WRITE_STRING
                           00000000G    EF  9F 00064              PUSHAB  PAS$FV_OUTPUT
                00000000G  EF              01   FB 0006A          CALLS   #1,PAS$WRITELN2
                           33334033     8F  DF 00071              PUSHAF  #^F0.7                          ; 1017
                00000000G  EF              01   FB 00077          CALLS   #1,LIB$WAIT
                                     0000V 31   00007E          BRW     12$
            00V00000000G   EF              30   E0 00081 8$:      BBS     #48,PAS$FV_INPUT,9$             ; 1027
                           00000000G    EF  9F 00089              PUSHAB  PAS$FV_INPUT
                00000000G  EF              01   FB 0008F          CALLS   #1,PAS$LOOK_AHEAD
            00V00000000G   EF              31   E0 00096 9$:      BBS     #49,PAS$FV_INPUT,11$
                           00000000G    EF  9F 0009E              PUSHAB  PAS$FV_INPUT                    ; 1031
                00000000G  EF              01   FB 000A4          CALLS   #1,PAS$RESET2
                                           00   DD 000AB          PUSHL   #0                              ; 1032
                                           00   DD 000AD          PUSHL   #0
                                           00   DD 000AF          PUSHL   #0
                           00B3804B     8F  DD 000B1              PUSHL   #11763787
                00000000G  EF              04   FB 000B7          CALLS   #4,LIB$SIGNAL
                           000000FF     8F  DD 000BE 11$:         PUSHL   #255                            ; 1036
                           00000000G    EF  9F 000C4              PUSHAB  PAS$FV_INPUT
                           00000000G    EF  9F 000CA              PUSHAB  INPUT_STRING
                00000000G  EF              03   FB 000D0          CALLS   #3,PAS$READ_STRING
                           00000000G    EF  9F 000D7              PUSHAB  PAS$FV_INPUT
                00000000G  EF              01   FB 000DD          CALLS   #1,PAS$READLN2
                           00000000G    EF  9F 000E4              PUSHAB  CRLF                            ; 1037
                                           02   DD 000EA          PUSHL   #2
                           00000000G    EF  9F 000EC              PUSHAB  PAS$FV_OUTPUT
                00000000G  EF              03   FB 000F2          CALLS   #3,PAS$WRITE_STRING
                           00000000G    EF  9F 000F9              PUSHAB  PAS$FV_OUTPUT
                00000000G  EF              01   FB 000FF          CALLS   #1,PAS$WRITELN2
                      F8   AD 010E00FF     8F   D0 00106          MOVL    #17694975,-8(FP)                ; 1038
                      FC   AD 00000000G    EF   9E 0010E          MOVAB   INPUT_STRING,-4(FP)
                                 F8   AD   9F 00116              PUSHAB  -8(FP)
                           00000000G    EF  9F 00119              PUSHAB  INPUT_DESC
                00000000G  EF              02   FB 0011F          CALLS   #2,STR$TRIM
                           00000000G    EF  9F 00126              PUSHAB  INPUT_DESC                      ; 1039
                           00000000G    EF  9F 0012C              PUSHAB  INPUT_DESC
                00000000G  EF              02   FB 00132          CALLS   #2,STR$UPCASE
            00V00000000G   EF              00   E1 00139 12$:      BBC     #0,JOURNAL_ENABLED,17$          ; 1047
                           00000000G    EF  B5 00141              TSTW    INPUT_DESC                      ; 1049
                                        00V 1B 00147              BLEQU   15$
                      7E   00000000G    EF   3C 00149             MOVZWL  INPUT_DESC,-(SP)                ; 1051
                                           00   DD 00150          PUSHL   #0
                      50   00000004G    EF   D0 00152             MOVL    INPUT_DESC+4,R0
                                           60   9F 00159          PUSHAB  (R0)
                           000000FF     8F  DD 0015B              PUSHL   #255
                           00000000G    EF  9F 00161              PUSHAB  JOURNAL_FILE
                00000000G  EF              05   FB 00167          CALLS   #5,PAS$WRITE_STRING
                           00000000G    EF  9F 0016E              PUSHAB  JOURNAL_FILE
                00000000G  EF              01   FB 00174          CALLS   #1,PAS$WRITELN2
                                        00V 11   00017B           BRB     16$
                           00000000G    EF  9F 0017D 15$:         PUSHAB  JOURNAL_FILE                    ; 1058
                00000000G  EF              01   FB 00183          CALLS   #1,PAS$WRITELN2
                                                0018A 16$:
                           00000000G    EF  B5 0018A 17$:         TSTW    INPUT_DESC                      ; 1063
                                        00V 12   00190            BNEQ    22$
                      00V                54   E9 00192          BLBC    DEFAULT_OK,20$                  ; 1065
                00000000G  EF              5C   D0 00195          MOVL    DEFAULT_VALUE,INPUT_VALUE       ; 1067
```

```
                                    0000V  31 0019C          BRW     28$
                                      00   DD 0019F 20$:     PUSHL   #0                                          ; 1073
                                      00   DD 001A1          PUSHL   #0
                                      00   DD 001A3          PUSHL   #0
                              00B38040     8F   DD 001A5          PUSHL   #11763776
                  00000000G   EF           04   FB 001AB          CALLS   #4,LIB$SIGNAL
                                      00V  11   001B2          BRB     28$
                  0000000CG   EF 00000004G EF   D0 001B4 22$:     MOVL    INPUT_DESC+4,PARAM_BLOCK+12               ; 1084
                  00000008G   EF 00000000G EF   3C 001BF          MOVZWL  INPUT_DESC,PARAM_BLOCK+8                  ; 1085
                                      52   DD 001CA          PUSHL   KEY_TABLE                                  ; 1087
                                      53   DD 001CC          PUSHL   STATE_TABLE
                              00000000G    EF   9F 001CE          PUSHAB  PARAM_BLOCK
                  00000000G   EF           03   FB 001D4          CALLS   #3,LIB$TPARSE
                  00000000G   EF           50   D0 001DB          MOVL    R0,ISTATUS
                  00000000G   EF 00000020G EF   D0 001E2          MOVL    PARAM_BLOCK+32,INPUT_VALUE                ; 1093
                  00000000G   EF 0000001CG EF   D0 001ED          MOVL    PARAM_BLOCK+28,INPUT_NUMBER               ; 1094
                              00V00000000G EF   E8 001F8          BLBS    ISTATUS,28$                               ; 1099
              00V00000006G    EF           00   E1 001FF          BBC     #0,PARAM_BLOCK+6,25$                      ; 1103
                                      00   DD 00207          PUSHL   #0                                         ; 1105
                                      00   DD 00209          PUSHL   #0
                                      00   DD 0020B          PUSHL   #0
                              00B38028     8F   DD 0020D          PUSHL   #11763752
                  00000000G   EF           04   FB 00213          CALLS   #4,LIB$SIGNAL
                                      00V  11   0021A          BRB     28$
                                      00   DD 0021C 25$:     PUSHL   #0                                         ; 1109
                                      00   DD 0021E          PUSHL   #0
                                      00   DD 00220          PUSHL   #0
                              00B38030     8F   DD 00222          PUSHL   #11763760
                  00000000G   EF           04   FB 00228          CALLS   #4,LIB$SIGNAL
                                           04   0022F 28$:     RET                                                ; 1115

; Routine Size: 560 bytes,    Routine Base: $CODE + 0067A

                                           00000 NUMBER_INPUT:                                                     ; 1161
                                      000C 00000          .WORD   ^M<R2,R3>
                          5E                08   C2 00002          SUBL2   #8,SP
                          52          08    BC   90 00005          MOVB    @8(R12),DEFAULT_OK
                          53          0C    BC   D0 00009          MOVL    @12(R12),DEFAULT_VALUE
              00000000G   EF 00000000G  EF    7D 0000D          MOVQ    NULL_STRING,INPUT_DESC                    ; 1172
          00V00000000G    EF                00   E1 00018          BBC     #0,TAKE_DEFAULTS,3$                       ; 1177
                              00000104G    EF   D5 00020          TSTL    IDATA+260
                                      00V  12   00026          BNEQ    3$
                          00V               52   E8 00028          BLBS    DEFAULT_OK,4$
          00V00000000G    EF                00   E1 0002B 3$:     BBC     #0,AUTO_TUNE,7$
          03 00000000G    EF                00   E1 00033 4$:     BBC     #0,AUTO_TUNE,.+3                          ; 1191
                                    0000V  31   00038          BRW     11$
                              00000000G    EF   9F 0003E          PUSHAB  CRLF                                      ; 1195
                                      02   DD 00044          PUSHL   #2
                              00000000G    EF   9F 00046          PUSHAB  PAS$FV_OUTPUT
                  00000000G   EF           03   FB 0004C          CALLS   #3,PAS$WRITE_STRING
                              00000000G    EF   9F 00053          PUSHAB  PAS$FV_OUTPUT
                  00000000G   EF           01   FB 00059          CALLS   #1,PAS$WRITELN2
                              33334033     8F   DF 00060          PUSHAF  #^F0.7                                    ; 1196
                  00000000G   EF           01   FB 00066          CALLS   #1,LIB$WAIT
                                    0000V  31   0006D          BRW     11$
          00V00000000G    EF                30   E0 00070 7$:     BBS     #48,PAS$FV_INPUT,8$                       ; 1206
                              00000000G    EF   9F 00078          PUSHAB  PAS$FV_INPUT
```

```
                                                
```

```
                                        04   AC    DD 001B5 21$:    PUSHL   4(R12)                                    ; 1265
                                 00000000G   EF    9F 001B8         PUSHAB  INPUT_DESC
                          00000000G  EF              02 FB 001BE    CALLS   #2,OTS$CVT_TI_L
                          00000000G  EF              50 D0 001C5    MOVL    R0,ISTATUS
                                  00V00000000G  EF    E8 001CC     BLBS    ISTATUS,24$                               ; 1270
                                                 00   DD 001D3     PUSHL   #0                                        ; 1274
                                                 00   DD 001D5     PUSHL   #0
                                                 00   DD 001D7     PUSHL   #0
                                        00B38030 8F   DD 001D9     PUSHL   #11763760
                          00000000G  EF              04   FB 001DF CALLS   #4,LIB$SIGNAL
                          00000000G  EF        04    BC   D0 001E6 24$:    MOVL    @4(R12),INPUT_VALUE               ; 1280
                          00000000G  EF        04    BC   D0 001EE         MOVL    @4(R12),INPUT_NUMBER              ; 1281
                                                 04   001F6         RET                                               ; 1283

; Routine Size: 503 bytes,    Routine Base: $CODE + 0C8AA

                                                 00000 MAKE_SCRATCH:                                                    ; 1329
                                              003C 00000           .WORD   ^M<R2,R3,R4,R5>
                                  5E        C0   AE   9E 00002     MOVAB   -64(SP),SP
                                        00000040 8F   DD 00006     PUSHL   #64                                       ; 1336
                          00000000G  EF              01   FB 0000C CALLS   #1,PAS$NEW2
                          00000000G  EF              50   D0 00013 MOVL    R0,DEF_SCRATCH
                                CO   AD 00000000G    EF   D0 0001A MOVL    DEF_SCRATCH,-64(FP)                       ; 1341
                     CO   BD 00000000G  EF     0040  8F   28 00022 MOVC3   #64,LINE_OBJECT_TEMPLATE,@-64(FP)
                                                 04   0002D         RET                                               ; 1343

; Routine Size: 46 bytes,    Routine Base: $CODE + 0AA1

                                                 00000 CURRENT_GT_TEST:                                                 ; 1390
                                              003C 00000           .WORD   ^M<R2,R3,R4,R5>
                                  5E        C0   AE   9E 00002     MOVAB   -64(SP),SP
                     CO   AD        04    BC     0040  8F   28 00006 MOVC3   #64,@4(R12),TEST
                                  5C          08   BC   90 0000E     MOVB    @8(R12),EXACT_COMPARISON
                                              50   94   00012       CLRB    CURRENT_GT_TEST                           ; 1397
                           51 00000000G  EF              19   C1 00014 ADDL3   #25,DEF_CURRENT,R1                    ; 1402
                                  51                    61   9A 0001C MOVZBL  (R1),R1
                                  51 00000000GEF        41   9E 0001F MOVAB   PRI_SEQ[R1],R1
                                  52          D9        AD   9A 00027 MOVZBL  TEST+25,R2
                                  52 00000000GEF        42   9E 0002B MOVAB   PRI_SEQ[R2],R2
                                  53 00000000G  EF              D0   00033 MOVL    DEF_CURRENT,R3
                                  53          1A        A3   D0 0003A MOVL    26(R3),R3
                                  00V                   5C   E9 0003E BLBC    EXACT_COMPARISON,16$
                                  62                    61   91 00041 CMPB    (R1),(R2)                             ; 1406
                                                        00V  1B 00044 BLEQU   3$
                                  50                    01   90 00046 MOVB    #1,CURRENT_GT_TEST                    ; 1410
                              DA  AD                    53   D1 00049 3$:    CMPL    R3,TEST+26                    ; 1412
                                                        00V  15 0004D BLEQ    6$
                                  62                    61   91 0004F CMPB    (R1),(R2)
                                                        00V  12 00052 BNEQ    6$
                                  50                    01   90 00054 MOVB    #1,CURRENT_GT_TEST                    ; 1418
                              DA  AD                    53   D1 00057 6$:    CMPL    R3,TEST+26                    ; 1420
                                                        00V  12 0005B BNEQ    10$
                                  62                    61   91 0005D CMPB    (R1),(R2)
                                                        00V  12 00060 BNEQ    10$
                                  5C 00000000G  EF              D0   00062 MOVL    DEF_CURRENT,R12
                              DF  AD          1F        AC   D1 00069 CMPL    31(R12),TEST+31
                                                        00V  15 0006E BLEQ    10$
```

```
                         50              01   90 00070          MOVB    #1,CURRENT_GT_TEST                      ; 1428
                         5C 00000000G    EF   D0 00073 10$:    MOVL    DEF_CURRENT,R12                         ; 1430
                      DE    AD      1E   AC   91 0007A          CMPB    30(R12),TEST+30
                                         00V  1B 0007F          BLEQU   22$
                      DA    AD           53   D1 00081          CMPL    R3,TEST+26
                                         00V  12 00085          BNEQ    22$
                         62              61   91 00087          CMPB    (R1),(R2)
                                         00V  12 0008A          BNEQ    22$
                         5C 00000000G    EF   D0 0008C          MOVL    DEF_CURRENT,R12
                      DF    AD      1F   AC   D1 00093          CMPL    31(R12),TEST+31
                                         00V  12 00098          BNEQ    22$
                         50              01   90 0009A          MOVB    #1,CURRENT_GT_TEST                      ; 1440
                                         00V  11 0009D          BRB     22$
                         62              61   91 0009F 16$:     CMPB    (R1),(R2)                               ; 1448
                                         00V  1B 000A2          BLEQU   18$
                         50              01   90 000A4          MOVB    #1,CURRENT_GT_TEST                      ; 1452
                      DA    AD           53   D1 000A7 18$:     CMPL    R3,TEST+26                              ; 1454
                                         00V  15 000AB          BLEQ    22$
                         62              61   91 000AD          CMPB    (R1),(R2)
                                         00V  12 000B0          BNEQ    22$
                         50              01   90 000B2          MOVB    #1,CURRENT_GT_TEST                      ; 1460
                                            04 000B5 22$:       RET                                             ; 1464

; Routine Size: 182 bytes,    Routine Base: $CODE + 00ACF

                                        00000 CURRENT_LT_TEST:                                                  ; 1511
                                     003C 00000          .WORD   ^M<R2,R3,R4,R5>
                         5E      C0   AE   9E 00002          MOVAB   -64(SP),SP
                   C0    AD      04   BC 0040 8F   28 00006  MOVC3   #64,@4(R12),TEST
                         5C      08   BC   90 0000E          MOVB    @8(R12),EXACT_COMPARISON
                                      50   94 00012          CLRB    CURRENT_LT_TEST                         ; 1518
                      51 00000000G    EF   19   C1 00014    ADDL3   #25,DEF_CURRENT,R1                      ; 1523
                         51           61   9A 0001C          MOVZBL  (R1),R1
                         51 00000000GEF41   9E 0001F          MOVAB   PRI_SEQ[R1],R1
                         52      D9   AD   9A 00027          MOVZBL  TEST+25,R2
                         52 00000000GEF42   9E 0002B          MOVAB   PRI_SEQ[R2],R2
                         53 00000000G    EF   D0 00033          MOVL    DEF_CURRENT,R3
                         53      1A   A3   D0 0003A          MOVL    26(R3),R3
                         00V          5C   E9 0003E          BLBC    EXACT_COMPARISON,16$
                         62           61   91 00041          CMPB    (R1),(R2)                               ; 1527
                                      00V  1E 00044          BGEQU   3$
                         50           01   90 00046          MOVB    #1,CURRENT_LT_TEST                      ; 1531
                      DA    AD      53   D1 00049 3$:        CMPL    R3,TEST+26                              ; 1533
                                      00V  18 0004D          BGEQ    6$
                         62           61   91 0004F          CMPB    (R1),(R2)
                                      00V  12 00052          BNEQ    6$
                         50           01   90 00054          MOVB    #1,CURRENT_LT_TEST                      ; 1539
                      DA    AD      53   D1 00057 6$:        CMPL    R3,TEST+26                              ; 1541
                                      00V  12 0005B          BNEQ    10$
                         62           61   91 0005D          CMPB    (R1),(R2)
                                      00V  12 00060          BNEQ    10$
                         5C 00000000G    EF   D0 00062          MOVL    DEF_CURRENT,R12
                      DF    AD      1F   AC   D1 00069          CMPL    31(R12),TEST+31
                                      00V  18 0006E          BGEQ    10$
                         50           01   90 00070          MOVB    #1,CURRENT_LT_TEST                      ; 1549
                         5C 00000000G    EF   D0 00073 10$:    MOVL    DEF_CURRENT,R12                         ; 1551
                      DE    AD      1E   AC   91 0007A          CMPB    30(R12),TEST+30
```

```
                                    00V  1E 0007F          BGEQU   22$
                              DA    AD   53   D1 00081     CMPL    R3,TEST+26
                                    00V  12 00085          BNEQ    22$
                                    62   61   91 00087     CMPB    (R1),(R2)
                                    00V  12 0008A          BNEQ    22$
                                    5C 00000000G EF   D0 0008C     MOVL    DEF_CURRENT,R12
                              DF    AD      1F    AC   D1 00093     CMPL    31(R12),TEST+31
                                    00V  12 00098          BNEQ    22$
                                    50   01   90 0009A     MOVB    #1,CURRENT_LT_TEST                      ; 1561
                                    00V  11 0009D          BRB     22$
                                    62   61   91 0009F 16$: CMPB    (R1),(R2)                           ; 1569
                                    00V  1E 000A2          BGEQU   18$
                                    50   01   90 000A4     MOVB    #1,CURRENT_LT_TEST                      ; 1573
                              DA    AD   53   D1 000A7 18$: CMPL    R3,TEST+26                          ; 1575
                                    00V  18 000AB          BGEQ    22$
                                    62   61   91 000AD     CMPB    (R1),(R2)
                                    00V  12 000B0          BNEQ    22$
                                    50   01   90 000B2     MOVB    #1,CURRENT_LT_TEST                      ; 1581
                                         04   000B5 22$:   RET                                          ; 1585

; Routine Size: 182 bytes,    Routine Base: $CODE + 00B85

                                         00000 CURRENT_EQ_TEST:                                         ; 1632
                                    003C 00000          .WORD   ^M<R2,R3,R4,R5>
                                    5E      C0   AE   9E 00002     MOVAB   -64(SP),SP                                  ;
                         C0   AD       04   BC   0040 8F   28 00006     MOVC3   #64,@4(R12),TEST
                                    5C      08   BC   90 0000E     MOVB    @8(R12),EXACT_COMPARISON
                              50 00000000G EF   19   C1 00012     ADDL3   #25,DEF_CURRENT,R0                  ; 1642
                                    60      D9   AD   91 0001A     CMPB    R1
                                                 00V  12 0001C     CMPB    TEST+25,(R0)
                                                 51   96 00020     BNEQ    2$
                                                 00V  12 00022     INCB    R1
                         50 00000000G EF         1A   C1 00024 2$: ADDL3   #26,DEF_CURRENT,R0
                                                 52   94 0002A     CLRB    R2
                                    60      DA   AD   D1 0002C     CMPL    TEST+26,(R0)
                                                 00V  12 00030     BNEQ    4$
                                                 52   96 00032     INCB    R2
                                    00V          5C   E9 00034 4$: BLBC    EXACT_COMPARISON,13$
                                    5C 00000000G EF   D0 00037     MOVL    DEF_CURRENT,R12                 ; 1644
                               1F   AC      DF   AD   D1 00040     CMPL    TEST+31,31(R12)
                                                 00V  12 00045     BNEQ    11$
                                    5C 00000000G EF   D0 00047     MOVL    DEF_CURRENT,R12
                               1E   AC      DE   AD   91 0004E     CMPB    TEST+30,30(R12)
                                                 00V  12 00053     BNEQ    11$
                                    00V          52   E9 00055     BLBC    R2,11$
                                    5C 00000000G EF   D0 00058     MOVL    DEF_CURRENT,R12
                                    6C      C0   AD   91 0005F     CMPB    TEST,(R12)
                                                 00V  12 00063     BNEQ    11$
                                    00V          51   E9 00065     BLBC    R1,11$
                                    5C           01   90 00068     MOVB    #1,CURRENT_EQ_TEST
                                                 00V  11 0006B     BRB     14$
                                                 5C   94 0006D 11$: CLRB    CURRENT_EQ_TEST
                                                 00V  11 0006F     BRB     14$
                                    52           52   92 00071 13$: MCOMB   R2,R2                           ; 1660
                         5C         51           52   8B 00074     BICB3   R2,R1,CURRENT_EQ_TEST
                                    50           5C   90 00078 14$: MOVB    CURRENT_EQ_TEST,R0              ; 1668
                                                 04   0007B       RET
```

```
; Routine Size: 124 bytes,    Routine Base: $CODE + 00C3B

                                    00000 INSERT_BEFORE_CURRENT:                                    ; 1716
                                0000 00000          .WORD   ^M<>
00000000G  EF 00000000G  EF   D1 00002          CMPL    DEF_CURRENT,DEF_HEAD                    ; 1723
                         00V  12 0000D          BNEQ    2$
00000000G  EF 00000000G  EF   D0 0000F          MOVL    DEF_SCRATCH,DEF_HEAD                    ; 1725
              50 00000000G  EF   D0 0001A 2$:   MOVL    DEF_CURRENT,R0                          ; 1730
00000000G  EF       05     A0   D0 00021          MOVL    5(R0),DEF_PRED
              50 00000000G  EF   D0 00029          MOVL    DEF_SCRATCH,R0                          ; 1731
      01      A0 00000000G  EF   D0 00030          MOVL    DEF_CURRENT,1(R0)
              50 00000000G  EF   D0 00038          MOVL    DEF_SCRATCH,R0                          ; 1732
      05      A0 00000000G  EF   D0 0003F          MOVL    DEF_PRED,5(R0)
                 00000000G  EF   D5 00047          TSTL    DEF_PRED                                ; 1734
                         00V  13 0004D          BEQL    4$
              50 00000000G  EF   D0 0004F          MOVL    DEF_PRED,R0                             ; 1736
      01      A0 00000000G  EF   D0 00056          MOVL    DEF_SCRATCH,1(R0)
              50 00000000G  EF   D0 0005E 4$:      MOVL    DEF_CURRENT,R0                          ; 1738
      05      A0 00000000G  EF   D0 00065          MOVL    DEF_SCRATCH,5(R0)
00000000G  EF 00000000G  EF   D0 0006D          MOVL    DEF_SCRATCH,DEF_CURRENT                 ; 1743
                                04 00078          RET                                             ; 1745

; Routine Size: 121 bytes,    Routine Base: $CODE + 00CB7

                                    00000 INSERT_AT_CURRENT:                                        ; 1793
                                0000 00000          .WORD   ^M<>
00000000G  EF 00000000G  EF   D1 00002          CMPL    DEF_CURRENT,DEF_HEAD                    ; 1800
                         00V  12 0000D          BNEQ    2$
00000000G  EF 00000000G  EF   D0 0000F          MOVL    DEF_SCRATCH,DEF_HEAD                    ; 1802
00000000G  EF 00000000G  EF   D1 0001A 2$:      CMPL    DEF_CURRENT,DEF_TAIL                    ; 1804
                         00V  12 00025          BNEQ    4$
00000000G  EF 00000000G  EF   D0 00027          MOVL    DEF_SCRATCH,DEF_TAIL                    ; 1806
              50 00000000G  EF   D0 00032 4$:   MOVL    DEF_CURRENT,R0                          ; 1811
00000000G  EF       05     A0   D0 00039          MOVL    5(R0),DEF_PRED
              50 00000000G  EF   D0 00041          MOVL    DEF_CURRENT,R0                          ; 1812
00000000G  EF       01     A0   D0 00048          MOVL    1(R0),DEF_SUCC
              50 00000000G  EF   D0 00050          MOVL    DEF_SCRATCH,R0                          ; 1813
              51 00000000G  EF   D0 00057          MOVL    DEF_CURRENT,R1
      01      A0       01   A1   D0 0005E          MOVL    1(R1),1(R0)
              50 00000000G  EF   D0 00063          MOVL    DEF_SCRATCH,R0                          ; 1814
              51 00000000G  EF   D0 0006A          MOVL    DEF_CURRENT,R1
      05      A0       05   A1   D0 00071          MOVL    5(R1),5(R0)
                 00000000G  EF   D5 00076          TSTL    DEF_PRED                                ; 1816
                         00V  13 0007C          BEQL    6$
              50 00000000G  EF   D0 0007E          MOVL    DEF_PRED,R0                             ; 1818
      01      A0 00000000G  EF   D0 00085          MOVL    DEF_SCRATCH,1(R0)
                 00000000G  EF   D5 0008D 6$:      TSTL    DEF_SUCC                                ; 1820
                         00V  13 00093          BEQL    8$
              50 00000000G  EF   D0 00095          MOVL    DEF_SUCC,R0                             ; 1822
      05      A0 00000000G  EF   D0 0009C          MOVL    DEF_SCRATCH,5(R0)
                 00000000G  EF   DD 000A4 8$:      PUSHL   DEF_CURRENT                             ; 1827
00000000G  EF                   01   FB 000AA          CALLS   #1,PAS$DISPOSE2
00000000G  EF 00000000G  EF   D0 000B1          MOVL    DEF_SCRATCH,DEF_CURRENT                 ; 1829
                                04 000BC          RET                                             ; 1831

; Routine Size: 189 bytes,    Routine Base: $CODE + 00D30
```

```
                                                00000 INSERT_AFTER_CURRENT:                                     ; 1879
                                           0000 00000         .WORD   ^M<>
00000000G  EF 00000000G  EF   D1 00002         CMPL    DEF_CURRENT,DEF_TAIL                                 ; 1886
                         00V  12 00000D         BNEQ    2$
00000000G  EF 00000000G  EF   D0 0000F         MOVL    DEF_SCRATCH,DEF_TAIL                                 ; 1888
              50 00000000G  EF   D0 0001A 2$:   MOVL    DEF_CURRENT,R0                                       ; 1893
00000000G  EF        01    A0   D0 00021         MOVL    1(R0),DEF_SUCC
              50 00000000G  EF   D0 00029         MOVL    DEF_SCRATCH,R0                                       ; 1894
        01    A0 00000000G  EF   D0 00030         MOVL    DEF_SUCC,1(R0)
              50 00000000G  EF   D0 00038         MOVL    DEF_SCRATCH,R0                                       ; 1895
        05    A0 00000000G  EF   D0 0003F         MOVL    DEF_CURRENT,5(R0)
              50 00000000G  EF   D0 00047         MOVL    DEF_CURRENT,R0                                       ; 1896
        01    A0 00000000G  EF   D0 0004E         MOVL    DEF_SCRATCH,1(R0)
                 00000000G  EF   D5 00056         TSTL    DEF_SUCC                                             ; 1898
                         00V  13 0005C         BEQL    4$
              50 00000000G  EF   D0 0005E         MOVL    DEF_SUCC,R0                                          ; 1900
        05    A0 00000000G  EF   D0 00065         MOVL    DEF_SCRATCH,5(R0)
00000000G  EF 00000000G  EF   D0 0006D 4$:      MOVL    DEF_SCRATCH,DEF_CURRENT                              ; 1905
                              04 00078         RET                                                          ; 1907

; Routine Size: 121 bytes,    Routine Base: $CODE + 00DED

                                                00000 INCR_CURRENT:                                             ; 1953
                                           0000 00000         .WORD   ^M<>
                 00000000G  EF   D5 00002         TSTL    DEF_CURRENT                                          ; 1960
                         00V  13 00008         BEQL    2$
              50 00000000G  EF   D0 0000A         MOVL    DEF_CURRENT,R0                                       ; 1962
00000000G  EF        01    A0   D0 00011         MOVL    1(R0),DEF_CURRENT
                              04 00019 2$:      RET                                                          ; 1964

; Routine Size: 26 bytes,    Routine Base: $CODE + 00E66

                                                00000 DECR_CURRENT:                                             ; 2010
                                           0000 00000         .WORD   ^M<>
                 00000000G  EF   D5 00002         TSTL    DEF_CURRENT                                          ; 2017
                         00V  13 00008         BEQL    2$
              50 00000000G  EF   D0 0000A         MOVL    DEF_CURRENT,R0                                       ; 2019
00000000G  EF        05    A0   D0 00011         MOVL    5(R0),DEF_CURRENT
                              04 00019 2$:      RET                                                          ; 2021

; Routine Size: 26 bytes,    Routine Base: $CODE + 00E80

                                                00000 NEW_IDENT_LINE:                                           ; 2075
                                           000C 00000         .WORD   ^M<R2,R3>
           5E                 1C   C2 00002         SUBL2   #28,SP
     0AA1  CF                 00   FB 00005         CALLS   #0,MAKE_SCRATCH                                      ; 2086
       F8  AD 010E0014        8F   D0 0000A         MOVL    #17694740,-8(FP)                                     ; 2091
       FC  AD        E4       AD   9E 00012         MOVAB   DATE_STRING,-4(FP)
             F8       AD       9F   9F 00017         PUSHAB  -8(FP)
00000000G  EF                 01   FB 0001A         CALLS   #1,LIB$DATE_TIME
           50                 01   D0 00021         MOVL    #1,R0                                                ; 2096
           5C                 50   D0 00024 2$:     MOVL    R0,I
FFFFFFFFGEF4C        E3 AD4C   90   9F 00027         MOVB    DATE_STRING-1[I],IDENT_STRING-1[I]                   ; 2098
EF         50                 14   F3 00031         AOBLEQ  #20,R0,2$
           52 00000000G       EF   D0 00035         MOVL    DEF_SCRATCH,R2                                       ; 2103
00000000G  EF 00000000G       EF   7D 0003C         MOVQ    NULL_STRING,TEMP_DESCRIPTOR                          ; 2107
```

```
                    000000FF   8F    DD 00047          PUSHL   #255                                    ; 2108
00000000G   EF                 01    FB 0004D          CALLS   #1,PAS$NEW2
00000004G   EF                 50    D0 00054          MOVL    R0,TEMP_DESCRIPTOR+4
00000000G   EF 00000000G       EF    B0 0005B          MOVW    IDENT_STRING_LENGTH,TEMP_DESCRIPTOR    ; 2110
                               62    94 00066          CLRB    (R2)                                   ; 2111
          19   A2              09    90 00068          MOVB    #9,25(R2)                              ; 2112
               50              01    D0 0006C          MOVL    #1,R0                                  ; 2114
               51 00000000G    EF    D0 0006F          MOVL    IDENT_STRING_LENGTH,R1
               51              50    D1 00076          CMPL    R0,R1
                               00V   15 00079          BLEQ    5$
                               00V   11 0007B          BRB     6$
                               50    D6 0007D 4$:      INCL    R0
               5C              50    D0 0007F 5$:      MOVL    R0,I
               53 00000004G    EF    D0 00082          MOVL    TEMP_DESCRIPTOR+4,R3                   ; 2116
        FF A34C FFFFFFFFGEF4C  90    90 00089          MOVB    IDENT_STRING-1[I],-1(R3)[I]
               51              50    D1 00093          CMPL    R0,R1
                               E5    19 00096          BLSS    4$
                      11       A2    9F 00098 6$:      PUSHAB  17(R2)                                 ; 2118
                    00000000G  EF    9F 0009B          PUSHAB  TEMP_DESCRIPTOR
00000000G   EF                 02    FB 000A1          CALLS   #2,LIB$SCOPY_DXDX
                    00000004G  EF    DD 000A8          PUSHL   TEMP_DESCRIPTOR+4                      ; 2119
00000000G   EF                 01    FB 000AE          CALLS   #1,PAS$DISPOSE2
00000000G   EF 00000000G       EF    D0 000B5          MOVL    DEF_SCRATCH,DEF_CURRENT                ; 2126
00000000G   EF 00000000G       EF    D0 000C0          MOVL    DEF_SCRATCH,DEF_HEAD                   ; 2127
00000000G   EF 00000000G       EF    D0 000CB          MOVL    DEF_SCRATCH,DEF_TAIL                   ; 2128
                               04    D6 000D6          RET                                            ; 2130

; Routine Size: 215 bytes,    Routine Base: $CODE + 00E9A

                                     00000 DELETE_CURRENT:                                            ; 2180
                               0000  00000             .WORD   ^M<>
               50 00000000G    EF    D0 00002          MOVL    DEF_CURRENT,R0                         ; 2184
               0F      19      A0    91 00009          CMPB    25(R0),#15
                               00V   12 0000D          BNEQ    5$
               50 00000000G    EF    D0 0000F          MOVL    DEF_CURRENT,R0                         ; 2191
                       01      A0    D5 00016          TSTL    1(R0)
                               00V   12 00019          BNEQ    3$
                    00000000G  EF    DD 0001B          PUSHL   DEF_CURRENT                            ; 2195
00000000G   EF                 01    FB 00021          CALLS   #1,PAS$DISPOSE2
    0E9A    CF                 00    FB 00028          CALLS   #0,NEW_IDENT_LINE                      ; 2196
                          0000V  31    2D 0002D          BRW     27$
               50 00000000G    EF    D0 00030 3$:      MOVL    DEF_CURRENT,R0                         ; 2204
00000000G   EF         01      A0    D0 00037          MOVL    1(R0),DEF_HEAD
               50 00000000G    EF    D0 0003F          MOVL    DEF_HEAD,R0                            ; 2205
                       05      A0    D4 00046          CLRL    5(R0)
                    00000000G  EF    DD 00049          PUSHL   DEF_CURRENT                            ; 2206
00000000G   EF                 01    FB 0004F          CALLS   #1,PAS$DISPOSE2
00000000G   EF 00000000G       EF    D0 00056          MOVL    DEF_HEAD,DEF_CURRENT                   ; 2207
                          0000V  31    2D 00061          BRW     27$
                    00000000G  EF    D5 00064 5$:      TSTL    DEF_CURRENT                            ; 2220
                               00V   13 0006A          BEQL    8$
00000000G   EF 00000000G       EF    D1 0006C          CMPL    DEF_CURRENT,DEF_TAIL
                               00V   12 00077          BNEQ    8$
               50 00000000G    EF    D0 00079          MOVL    DEF_CURRENT,R0                         ; 2222
00000000G   EF         05      A0    D0 00080          MOVL    5(R0),DEF_TAIL
                    00000000G  EF    D5 00088 8$:      TSTL    DEF_CURRENT                            ; 2227
                               00V   13 0008E          BEQL    11$
```

```
00000000G  EF 00000000G  EF   D1 00090          CMPL    DEF_CURRENT,DEF_HEAD
                         00V  12 0009B          BNEQ    11$
               50 00000000G  EF   D0 0009D      MOVL    DEF_CURRENT,R0                          ; 2229
00000000G  EF        01      A0   D0 000A4      MOVL    1(R0),DEF_HEAD
               50 00000000G  EF   D0 000AC 11$: MOVL    DEF_CURRENT,R0                          ; 2234
00000000G  EF        05      A0   D0 000B3      MOVL    5(R0),DEF_PRED
               50 00000000G  EF   D0 000BB      MOVL    DEF_CURRENT,R0                          ; 2235
00000000G  EF        01      A0   D0 000C2      MOVL    1(R0),DEF_SUCC
                  00000000G  EF   D5 000CA      TSTL    DEF_PRED                                ; 2237
                         00V  13 000D0          BEQL    13$
               50 00000000G  EF   D0 000D2      MOVL    DEF_PRED,R0                             ; 2239
        01     A0 00000000G  EF   D0 000D9      MOVL    DEF_SUCC,1(R0)
                  00000000G  EF   D5 000E1 13$: TSTL    DEF_SUCC                                ; 2241
                         00V  13 000E7          BEQL    15$
               50 00000000G  EF   D0 000E9      MOVL    DEF_SUCC,R0                             ; 2243
        05     A0 00000000G  EF   D0 000F0      MOVL    DEF_PRED,5(R0)
               5C 00000000G  EF   D0 000F8 15$: MOVL    DEF_CURRENT,R12                         ; 2245
                        11     AC   B5 000FF      TSTW    17(R12)                                 ; 2249
                         00V  15 00102          BLEQ    18$
                        11     AC   9F 00104      PUSHAB  17(R12)                                 ; 2251
00000000G  EF                  01   FB 00107      CALLS   #1,STR$FREE1_DX
                        09     AC   B5 0010E 18$: TSTW    9(R12)                                  ; 2253
                         00V  15 00111          BLEQ    20$
                        09     AC   9F 00113      PUSHAB  9(R12)                                  ; 2255
00000000G  EF                  01   FB 00116      CALLS   #1,STR$FREE1_DX
                  00000000G  EF   DD 0011D 20$: PUSHL   DEF_CURRENT                             ; 2259
00000000G  EF                  01   FB 00123      CALLS   #1,PAS$DISPOSE2
                  00000000G  EF   D5 0012A      TSTL    DEF_SUCC                                ; 2261
                         00V  13 00130          BEQL    22$
00000000G  EF 00000000G  EF   D0 00132          MOVL    DEF_SUCC,DEF_CURRENT                    ; 2263
                         00V  11 0013D          BRB     26$
                  00000000G  EF   D5 0013F 22$: TSTL    DEF_PRED                                ; 2265
                         00V  13 00145          BEQL    24$
00000000G  EF 00000000G  EF   D0 00147          MOVL    DEF_PRED,DEF_CURRENT                    ; 2267
                         00V  11 00152          BRB     25$
           0E9A  CF              00   FB 00154 24$: CALLS   #0,NEW_IDENT_LINE                    ; 2271
                                      00159 25$:
                                      00159 26$:
                                04   00159 27$: RET                                             ; 2275

; Routine Size: 346 bytes,    Routine Base: $CODE + 00F71

                                      00000 DELETE_PRIMARY_SECTION:                             ; 2327
                                001C  00000          .WORD   ^M<R2,R3,R4>
               52        04     BC   90 00002      MOVB    @4(R12),WHICHPRIMARY
               5C        08     BC   D0 00006      MOVL    @8(R12),WHICHPRINUM
00000000G  EF 00000000G  EF   D0 0000A          MOVL    DEF_HEAD,DEF_CURRENT                    ; 2336
                         53   94 00015          CLRB    DOING                                   ; 2337
                         54   94 00017          CLRB    DONE                                    ; 2338
               50 00000000G  EF   D0 00019 1$:  MOVL    DEF_CURRENT,R0                          ; 2348
               5C        1A     A0   D1 00020      CMPL    26(R0),WHICHPRINUM
                         00V  12 00024          BNEQ    5$
               50 00000000G  EF   D0 00026      MOVL    DEF_CURRENT,R0
                         60   95 0002D          TSTB    (R0)
                         00V  12 0002F          BNEQ    5$
               50 00000000G  EF   D0 00031      MOVL    DEF_CURRENT,R0
               52        19     A0   91 00038      CMPB    25(R0),WHICHPRIMARY
```

```
                                 00V  12 0003C          BNEQ    5$
                          53     01   90 0003E          MOVB    #1,DOING                                ; 2356
                          00V    53   E9 00041 5$:      BLBC    DOING,7$                                ; 2361
                    0F71  CF            00   FB 00044          CALLS   #0,DELETE_CURRENT               ; 2363
                                 00V  11 00049          BRB     8$
                    0E66  CF            00   FB 0004B 7$:      CALLS   #0,INCR_CURRENT                 ; 2370
                             00000000G  EF   D5 00050 8$:      TSTL    DEF_CURRENT                     ; 2376
                                 00V  13 00056          BEQL    15$
                          00V    53   E9 00058          BLBC    DOING,15$                               ; 2378
                          50 00000000G  EF   D0 0005B          MOVL    DEF_CURRENT,R0
                                        60   95 00062          TSTB    (R0)
                                 00V  12 00064          BNEQ    15$
                          50 00000000G  EF   D0 00066          MOVL    DEF_CURRENT,R0
                          5C        1A  A0   D1 0006D          CMPL    26(R0),WHICHPRINUM
                                 00V  12 00071          BNEQ    13$
                          50 00000000G  EF   D0 00073          MOVL    DEF_CURRENT,R0
                          52        19  A0   91 0007A          CMPB    25(R0),WHICHPRIMARY
                                 00V  13 0007E          BEQL    15$
                          54     01   90 00080 13$:      MOVB    #1,DONE                                 ; 2390
                          00V    54   E8 00083 15$:      BLBS    DONE,17$
                             00000000G  EF   D5 00086          TSTL    DEF_CURRENT
                                 8B   12 0008C          BNEQ    1$
                                      04 0008E 17$:      RET                                             ; 2394

; Routine Size: 143 bytes,    Routine Base: $CODE + 010CB

                                      00000 INIT_DEF:                                                   ; 2441
                                 0000 00000          .WORD   ^M<>
                    00000000G  EF 00000000G  EF   D0 00002          MOVL    DEF_HEAD,DEF_CURRENT        ; 2448
                                 00V  13 0000D          BEQL    6$                                      ; 2450
                       0F71  CF            00   FB 0000F 2$:      CALLS   #0,DELETE_CURRENT               ; 2456
                    00000000G  EF 00000000G  EF   D1 00014          CMPL    DEF_HEAD,DEF_TAIL
                                 EE   12 0001F          BNEQ    2$
                             00000000G  EF   D5 00021          TSTL    DEF_CURRENT                     ; 2460
                                 00V  13 00027          BEQL    6$
                       0F71  CF            00   FB 00029          CALLS   #0,DELETE_CURRENT               ; 2462
                                      04 0002E 6$:       RET                                             ; 2466

; Routine Size: 47 bytes,    Routine Base: $CODE + 0115A

                                      00000 INSERT_IN_ORDER:                                            ; 2516
                                 001C 00000          .WORD   ^M<R2,R3,R4>
                          5C        04   BC   D0 00002          MOVL    @4(R12),COLLISION_ACTION
                          50 00000000G  EF   D0 00006          MOVL    DEF_SCRATCH,R0                  ; 2528
                                        60   95 0000D          TSTB    (R0)
                                 00V  12 0000F          BNEQ    3$
                          50 00000000G  EF   D0 00011          MOVL    DEF_SCRATCH,R0
                          0F        19  A0   91 00018          CMPB    25(R0),#15
                                 00V  13 0001C          BEQL    3$
                          50 00000000G  EF   D0 0001E          MOVL    DEF_SCRATCH,R0                  ; 2534
                                    11  A0   B4 00025          CLRW    17(R0)
                          50 00000000G  EF   D0 00028 3$:      MOVL    DEF_SCRATCH,R0                  ; 2536
                                    01  A0   D4 0002F          CLRL    1(R0)
                          50 00000000G  EF   D0 00032          MOVL    DEF_SCRATCH,R0                  ; 2537
                                    05  A0   D4 00039          CLRL    5(R0)
                                        52   94 0003C          CLRB    BACKUP_WORKED                   ; 2542
                             00000000G  EF   D5 0003E          TSTL    DEF_CURRENT                     ; 2544
```

```
                    03   12 00044          BNEQ    .+3
                0000V    31 00046          BRW     14$
      50 00000000G  EF   D0 00049          MOVL    DEF_CURRENT,R0                          ; 2546
              05    A0   D5 00050          TSTL    5(R0)
                    03   12 00053          BNEQ    .+3
                0000V    31 00055          BRW     14$
0E80  CF              00   FB 00058          CALLS   #0,DECR_CURRENT                         ; 2550
                    00V  11 0005D          BRB     7$                                      ; 2552
0E66  CF              00   FB 0005F 6$:      CALLS   #0,INCR_CURRENT                         ; 2560
              01      8F   9F 00064 7$:      PUSHAB  #1
      50 00000000G    EF   D0 00067          MOVL    DEF_SCRATCH,R0
                      60   9F 0006E          PUSHAB  (R0)
0ACF  CF              02   FB 00070          CALLS   #2,CURRENT_GT_TEST
      53              50   90 00075          MOVB    R0,R3
              01      8F   9F 00078          PUSHAB  #1
      50 00000000G    EF   D0 0007B          MOVL    DEF_SCRATCH,R0
                      60   9F 00082          PUSHAB  (R0)
0C3B  CF              02   FB 00084          CALLS   #2,CURRENT_EQ_TEST
      50              53   88 00089          BISB2   R3,R0
                      53   94 0008C          CLRB    R3
      51 00000000G    EF   D0 0008E          MOVL    DEF_CURRENT,R1
              01      A1   D5 00095          TSTL    1(R1)
                    00V  12 00098          BNEQ    9$
                      53   96 0009A          INCB    R3
      53              50   88 0009C 9$:      BISB2   R0,R3
      BD              53   E9 0009F          BLBC    R3,6$
              01      8F   9F 000A2          PUSHAB  #1                                      ; 2562
      50 00000000G    EF   D0 000A5          MOVL    DEF_SCRATCH,R0
                      60   9F 000AC          PUSHAB  (R0)
0C3B  CF              02   FB 000AE          CALLS   #2,CURRENT_EQ_TEST
      53              50   90 000B3          MOVB    R0,R3
              01      8F   9F 000B6          PUSHAB  #1
      50 00000000G    EF   D0 000B9          MOVL    DEF_SCRATCH,R0
                      60   9F 000C0          PUSHAB  (R0)
        0B85  CF      02   FB 000C2          CALLS   #2,CURRENT_LT_TEST
                      51   94 000C7          CLRB    R1
      54 00000000G    EF   D0 000C9          MOVL    DEF_CURRENT,R4
              01      A4   D5 000D0          TSTL    1(R4)
                    00V  13 000D3          BEQL    12$
                      51   96 000D5          INCB    R1
      50              51   8A 000D7 12$:      BICB2   R1,R0
52    53              50   89 000DA          BISB3   R0,R3,BACKUP_WORKED
      00V             52   E8 000DE 14$:      BLBS    BACKUP_WORKED,21$                       ; 2574
  00000000G EF 00000000G    EF   D0 000E1          MOVL    DEF_HEAD,DEF_CURRENT            ; 2581
                    00V  11 000EC          BRB     17$                                     ; 2583
        0E66  CF      00   FB 000EE 16$:      CALLS   #0,INCR_CURRENT                         ; 2591
              01      8F   9F 000F3 17$:      PUSHAB  #1
      52 00000000G    EF   D0 000F6          MOVL    DEF_SCRATCH,R2
                      62   9F 000FD          PUSHAB  (R2)
        0ACF  CF      02   FB 000FF          CALLS   #2,CURRENT_GT_TEST
      52              50   90 00104          MOVB    R0,R2
              01      8F   9F 00107          PUSHAB  #1
      50 00000000G    EF   D0 0010A          MOVL    DEF_SCRATCH,R0
                      60   9F 00111          PUSHAB  (R0)
        0C3B  CF      02   FB 00113          CALLS   #2,CURRENT_EQ_TEST
      50              52   88 00118          BISB2   R2,R0
                      52   94 0011B          CLRB    R2
```

```
                53 00000000G   EF    D0 0011D          MOVL    DEF_CURRENT,R3
                      01       A3    D5 00124          TSTL    1(R3)
                               00V   12 00127          BNEQ    19$
                               52    96 00129          INCB    R2
                52             50    88 0012B 19$:     BISB2   R0,R2
                BD             52    E9 0012E          BLBC    R2,16$
                         01    8F    9F 00131 21$:     PUSHAB  #1                              ; 2598
                50 00000000G   EF    D0 00134          MOVL    DEF_SCRATCH,R0
                               60    9F 0013B          PUSHAB  (R0)
          0ACF  CF             02    FB 0013D          CALLS   #2,CURRENT_GT_TEST
                00V            50    E9 00142          BLBC    R0,23$
          0CB7  CF             00    FB 00145          CALLS   #0,INSERT_BEFORE_CURRENT        ; 2600
                               00V   11 0014A          BRB     34$
                         01    8F    9F 0014C 23$:     PUSHAB  #1                              ; 2602
                50 00000000G   EF    D0 0014F          MOVL    DEF_SCRATCH,R0
                               60    9F 00156          PUSHAB  (R0)
          0C3B  CF             02    FB 00158          CALLS   #2,CURRENT_EQ_TEST
                00V            50    E9 0015D          BLBC    R0,30$
                               5C    D5 00160          TSTL    COLLISION_ACTION                ; 2606
                               00V   12 00162          BNEQ    26$
          0D30  CF             00    FB 00164          CALLS   #0,INSERT_AT_CURRENT            ; 2608
                               00V   11 00169          BRB     34$
                02             5C    D1 0016B 26$:     CMPL    COLLISION_ACTION,#2             ; 2610
                               00V   12 0016E          BNEQ    34$
          0DED  CF             00    FB 00170          CALLS   #0,INSERT_AFTER_CURRENT         ; 2612
                               00v   11 00175          BRB     34$
                50 00000000G   EF    D0 00177 30$:     MOVL    DEF_CURRENT,R0                  ; 2618
                         01    A0    D5 0017E          TSTL    1(R0)
                               00V   12 00181          BNEQ    34$
    00000000G   EF 00000000G   EF    D0 00183          MOVL    DEF_CURRENT,DEF_TAIL            ; 2622
          0DED  CF             00    FB 0018E          CALLS   #0,INSERT_AFTER_CURRENT         ; 2623
                               04 00193 34$:     RET                                     ; 2627

; Routine Size: 404 bytes,    Routine Base: $CODE + 01189


                                     00000 FIND_OBJECT:                                    ; 2678
                                000C 00000          .WORD   ^M<R2,R3>
                      5E       C0    AE    9E 00002          MOVAB   -64(SP),SP
                      50       04    BC    90 00006          MOVB    @4(R12),OBJ_TYP
                      51       08    BC    90 0000A          MOVB    @8(R12),PRIM
                      52       0C    BC    D0 0000E          MOVL    @12(R12),PRIMNUM
                      53       10    BC    90 00012          MOVB    @16(R12),SECO
                      5C       14    BC    D0 00016          MOVL    @20(R12),SECONUM
                   C0    AD             50    90 0001A          MOVB    OBJ_TYP,TEST            ; 2696
                   D9    AD             51    90 0001E          MOVB    PRIM,TEST+25            ; 2697
                   DA    AD             52    D0 00022          MOVL    PRIMNUM,TEST+26         ; 2698
                   DE    AD             53    90 00026          MOVB    SECO,TEST+30            ; 2699
                   DF    AD             5C    D0 0002A          MOVL    SECONUM,TEST+31         ; 2700
    00000000G   EF 00000000G   EF    D0 0002E          MOVL    DEF_HEAD,DEF_CURRENT            ; 2705
                               5C    94 00039          CLRB    FOUND_IT                        ; 2706
                               53    94 0003B          CLRB    PAST_IT                         ; 2707
                00000000G      EF    D5 0003D          TSTL    DEF_CURRENT                     ; 2709
                               00V   13 00043          BEQL    8$
                         01    8F    9F 00045 2$:      PUSHAB  #1                              ; 2715
                         C0    AD    9F 00048          PUSHAB  TEST
          0C3B  CF             02    FB 0004B          CALLS   #2,CURRENT_EQ_TEST
                5C             50    90 00050          MOVB    R0,FOUND_IT
```

```
                                01   8F   9F 00053          PUSHAB  #1                                      ; 2716
                                C0   AD   9F 00056          PUSHAB  TEST
                          0ACF  CF        02  FB 00059      CALLS   #2,CURRENT_GT_TEST
                                53        50  90 0005E      MOVB    R0,PAST_IT
                                00V       5C  E8 00061      BLBS    FOUND_IT,4$                             ; 2718
                          0E66  CF        00  FB 00064      CALLS   #0,INCR_CURRENT                         ; 2720
                                00V       5C  E8 00069 4$:  BLBS    FOUND_IT,8$
                                00V       53  E8 0006C      BLBS    PAST_IT,8$
                                   00000000G  EF  D5 0006F      TSTL    DEF_CURRENT
                                              CE  12 00075      BNEQ    2$
                                50            5C  90 00077 8$:  MOVB    FIND_OBJECT,R0                        ; 2731
                                              04  00079      RET

; Routine Size: 123 bytes,    Routine Base: $CODE + 0131D

                                                  0000 00000      .ENTRY  POINT_AT_DEFINITION,^M<>          ; 2781
                   00V00000000G  EF        00     E0 00002      BBS     #0,POINTING_AT_DEFINITION,2$      ; 2785
                      00000000G  EF  00000000G EF  D0 0000A      MOVL    DEF_HEAD,DEF_ANL_HEAD             ; 2789
                      00000000G  EF  00000000G EF  D0 00015      MOVL    DEF_TAIL,DEF_ANL_TAIL             ; 2790
                      00000000G  EF  00000000G EF  D0 00020      MOVL    DEF_SAVE_HEAD,DEF_HEAD            ; 2791
                      00000000G  EF  00000000G EF  D0 0002B      MOVL    DEF_SAVE_TAIL,DEF_TAIL            ; 2792
                      00000000G  EF        01    90 00036      MOVB    #1,POINTING_AT_DEFINITION         ; 2794
                                              04  00039 2$:     RET                                       ; 2798

; Routine Size: 62 bytes,    Routine Base: $CODE + 01398

                                                  00000 POINT_AT_ANALYSIS:                                ; 2848
                                                  0000 00000      .WORD   ^M<>
                   00V00000000G  EF        00     E1 00002      BBC     #0,POINTING_AT_DEFINITION,2$      ; 2852
                      00000000G  EF  00000000G EF  D0 0000A      MOVL    DEF_HEAD,DEF_SAVE_HEAD            ; 2856
                      00000000G  EF  00000000G EF  D0 00015      MOVL    DEF_TAIL,DEF_SAVE_TAIL            ; 2857
                      00000000G  EF  00000000G EF  D0 00020      MOVL    DEF_ANL_HEAD,DEF_HEAD             ; 2858
                      00000000G  EF  00000000G EF  D0 0002B      MOVL    DEF_ANL_TAIL,DEF_TAIL             ; 2859
                                     00000000G EF  94 00036      CLRB    POINTING_AT_DEFINITION            ; 2861
                                               04  00038 2$:     RET                                       ; 2865

; Routine Size: 61 bytes,    Routine Base: $CODE + 013D6

                                                  0004 00000      .ENTRY  EDF$LINE_PARSED,^M<R2>            ; 2915
                                5C                01   D0 00002      MOVL    #1,EDF$LINE_PARSED          ; 2922
                                      0AA1  CF        00   FB 00005      CALLS   #0,MAKE_SCRATCH             ; 2927
                                50 00000000G  EF        D0 0000A      MOVL    FDL_BLOCK,R0                ; 2932
00000000G  EF                   18            00        60 F0 00011      INSV    (R0),#0,#24,TEMP_FDL3$TYPE
                   00V00000000G  EF        05   E1 0001A      BBC     #5,TEMP_FDL3$TYPE,2$                ; 2937
                                50 00000000G  EF   D0 00022      MOVL    FDL_BLOCK,R0
                                09       08   A0   91 00029      CMPB    8(R0),#9
                                              03   12 0002D      BNEQ    .+3
                                           0000V 31   0002F      BRW     76$
                   03 00000000G  EF        03   E1 00032 2$:   BBC     #3,TEMP_FDL3$TYPE,.+3
                                           0000V 31   0003A      BRW     76$
                                52 00000000G  EF   D0 0003D      MOVL    DEF_SCRATCH,R2                      ; 2945
                   00V00000000G  EF        05   E1 00044      BBC     #5,TEMP_FDL3$TYPE,6$                ; 2952
                                              62   94 0004C      CLRB    (R2)                                ; 2954
                                           00V   11 0004E      BRB     7$
                                62            01   90 00050 6$:   MOVB    #1,(R2)                         ; 2958
                   00V00000000G  EF        09   E1 00053 7$:   BBC     #9,TEMP_FDL3$TYPE,9$                ; 2963
                                62            02   90 0005B      MOVB    #2,(R2)                         ; 2965
```

```
                50 00000000G   EF    D0 0005E 9$:     MOVL    FDL_BLOCK,R0                            ; 2970
             19 A2        08   A0    90 00065         MOVB    8(R0),25(R2)
                50 00000000G   EF    D0 0006A         MOVL    FDL_BLOCK,R0                            ; 2971
             1E A2        14   A0    90 00071         MOVB    20(R0),30(R2)
00V00000001G EF                 00    E0 00076         BBS     #0,TEMP_FDL3$TYPE+1,11$                 ; 2976
00V00000000G EF                 09    E1 0007E         BBC     #9,TEMP_FDL3$TYPE,13$
   00000000G EF 00000000G       EF    7D 00086 11$:    MOVQ    NULL_STRING,TEMP_DESCRIPTOR             ; 2980
                50 00000000G   EF    D0 00091         MOVL    FDL_BLOCK,R0                            ; 2981
                        6C     A0    9F 00098         PUSHAB  108(R0)
                51 00000000G   EF    D0 0009B         MOVL    FDL_BLOCK,R1
                        68     A1    9F 000A2         PUSHAB  104(R1)
          04C1  CF                    02    FB 000A5         CALLS   #2,CVT_QUAD_DESC
   00000000G  EF                    50    7D 000AA         MOVQ    R0,TEMP_DESCRIPTOR
                        09     A2    9F 000B1         PUSHAB  9(R2)                                   ; 2985
                00000000G      EF    9F 000B4         PUSHAB  TEMP_DESCRIPTOR
   00000000G    EF                  02    FB 000BA         CALLS   #2,LIB$SCOPY_DXDX
00V00000000G    EF                  05    E1 000C1 13$:    BBC     #5,TEMP_FDL3$TYPE,15$           ; 2992
                0F      19     A2    91 000C9         CMPB    25(R2),#15
                               00V   13 000CD         BEQL    19$
00V00000000G    EF                  05    E0 000CF 15$:    BBS     #5,TEMP_FDL3$TYPE,17$
                21      1E     A2    91 000D7         CMPB    30(R2),#33
                               00V   13 000DB         BEQL    19$
00V00000000G    EF                  05    E0 000DD 17$:    BBS     #5,TEMP_FDL3$TYPE,21$
                50      1E     A2    9A 000E5         MOVZBL  30(R2),R0
                50             04    C4 000E9         MULL2   #4,R0
00V00000000G    EF                  50    E1 000EC         BBC     R0,SEC_TYPE,21$
   00000000G    EF 00000000G   EF    7D 000F4 19$:    MOVQ    NULL_STRING,TEMP_DESCRIPTOR             ; 3008
                50 00000000G   EF    D0 000FF         MOVL    FDL_BLOCK,R0                            ; 3009
                        64     A0    9F 00106         PUSHAB  100(R0)
                51 00000000G   EF    D0 00109         MOVL    FDL_BLOCK,R1
                        60     A1    9F 00110         PUSHAB  96(R1)
          04C1  CF                    02    FB 00113         CALLS   #2,CVT_QUAD_DESC
   00000000G  EF                    50    7D 00118         MOVQ    R0,TEMP_DESCRIPTOR
                        11     A2    9F 0011F         PUSHAB  17(R2)                                  ; 3013
                00000000G      EF    9F 00122         PUSHAB  TEMP_DESCRIPTOR
   00000000G    EF                  02    FB 00128         CALLS   #2,LIB$SCOPY_DXDX
                50      19     A2    9A 0012F 21$:    MOVZBL  25(R2),R0                               ; 3020
                10             50    D1 00133         CMPL    R0,#16
                               00V   1E 00136         BGEQU   23$
00VFFFFEB31     EF                  50    E1 00138         BBC     R0,C.AAH,23$
                50 00000000G   EF    D0 00140         MOVL    FDL_BLOCK,R0                            ; 3022
             1A A2        0C   A0    D0 00147         MOVL    12(R0),26(R2)
                               00V   11 0014C         BRB     24$
                        1A     A2    D4 0014E 23$:    CLRL    26(R2)                                  ; 3026
                50      1E     A2    9A 00151 24$:    MOVZBL  30(R2),R0                               ; 3028
   00000098     8F                  50    D1 00155         CMPL    R0,#152
                               00V   1E 0015C         BGEQU   26$
00VFFFFEB0F     EF                  50    E1 0015E         BBC     R0,C.AAI,26$
                50 00000000G   EF    D0 00166         MOVL    FDL_BLOCK,R0                            ; 3030
             1F A2        18   A0    D0 0016D         MOVL    24(R0),31(R2)
                               00V   11 00172         BRB     30$
             87 8F        1E   A2    91 00174 26$:    CMPB    30(R2),#-121                            ; 3037
                               00V   12 00179         BNEQ    28$
             1F A2                07    D0 0017B         MOVL    #7,31(R2)                               ; 3039
                               00V   11 0017F         BRB     29$
                        1F     A2    D4 00181 28$:    CLRL    31(R2)                                  ; 3043
                                     00184 29$:
```

```
                                Generated Code

                59  8F      1E  A2   91 00184 30$:    CMPB    30(R2),#89                      ; 3063
                                00V  13 00189        BEQL    32$
                83  8F      1E  A2   91 0018B        CMPB    30(R2),#-125
                                00V  12 00190        BNEQ    33$
                        23      A2   D4 00192 32$:    CLRL    35(R2)                          ; 3074
                    50 00000000G EF   D0 00195        MOVL    FDL_BLOCK,R0                    ; 3075
                27  A2      34  A0   D0 0019C        MOVL    52(R0),39(R2)
                            0000V  31 001A1        BRW     63$
                    50      1E  A2   9A 001A4 33$:    MOVZBL  30(R2),R0                       ; 3086
            29  62  8F          50   8F 001A8        CASEB   R0,#98,#41
                            0000V      001AD        .DISPL  34$
                            0054       001AF        .DISPL  84
                            0054       001B1        .DISPL  84
                            0054       001B3        .DISPL  84
                            0054       001B5        .DISPL  84
                            0054       001B7        .DISPL  84
                            0054       001B9        .DISPL  84
                            0054       001BB        .DISPL  84
                            0054       001BD        .DISPL  84
                            0054       001BF        .DISPL  84
                            0054       001C1        .DISPL  84
                            0054       001C3        .DISPL  84
                            0054       001C5        .DISPL  84
                            0054       001C7        .DISPL  84
                            0054       001C9        .DISPL  84
                            0054       001CB        .DISPL  84
                            0054       001CD        .DISPL  84
                            0054       001CF        .DISPL  84
                            0054       001D1        .DISPL  84
                            0054       001D3        .DISPL  84
                            0054       001D5        .DISPL  84
                            0054       001D7        .DISPL  84
                            0054       001D9        .DISPL  84
                            0054       001DB        .DISPL  84
                            0054       001DD        .DISPL  84
                            0054       001DF        .DISPL  84
                            0054       001E1        .DISPL  84
                            0054       001E3        .DISPL  84
                            0054       001E5        .DISPL  84
                            0054       001E7        .DISPL  84
                            0054       001E9        .DISPL  84
                            0054       001EB        .DISPL  84
                            0054       001ED        .DISPL  84
                            0054       001EF        .DISPL  84
                            0054       001F1        .DISPL  84
                            0054       001F3        .DISPL  84
                            0054       001F5        .DISPL  84
                            0000V      001F7        .DISPL  50$
                            0054       001F9        .DISPL  84
                            0054       001FB        .DISPL  84
                            0054       001FD        .DISPL  84
                            0000V      001FF        .DISPL  40$
                            0000V  31  00201        BRW     61$
                    50 00000000G EF   D0 00204 34$:    MOVL    FDL_BLOCK,R0                    ; 3090
            20      00      34  A0   CF 0020B        CASEL   52(R0),#0,#32
                            0000V      00210        .DISPL  37$
                            0042       00212        .DISPL  66
```

```
                                0042      00214          .DISPL  66
                                0042      00216          .DISPL  66
                                0042      00218          .DISPL  66
                                0042      0021A          .DISPL  66
                                0042      0021C          .DISPL  66
                                0042      0021E          .DISPL  66
                                0042      00220          .DISPL  66
                                0042      00222          .DISPL  66
                                0042      00224          .DISPL  66
                                0042      00226          .DISPL  66
                                0042      00228          .DISPL  66
                                0042      0022A          .DISPL  66
                                0042      0022C          .DISPL  66
                                0042      0022E          .DISPL  66
                                0000V     00230          .DISPL  36$
                                0042      00232          .DISPL  66
                                0042      00234          .DISPL  66
                                0042      00236          .DISPL  66
                                0042      00238          .DISPL  66
                                0042      0023A          .DISPL  66
                                0042      0023C          .DISPL  66
                                0042      0023E          .DISPL  66
                                0042      00240          .DISPL  66
                                0042      00242          .DISPL  66
                                0042      00244          .DISPL  66
                                0042      00246          .DISPL  66
                                0042      00248          .DISPL  66
                                0042      0024A          .DISPL  66
                                0042      0024C          .DISPL  66
                                0042      0024E          .DISPL  66
                                0000V     00250          .DISPL  35$
                                  00V  11 00252          BRB     38$
                          23 A2   1F  D0 00254 35$:      MOVL    #31,35(R2)              ; 3092
                                0000V 31 00258          BRW     62$
                          23 A2   1E  D0 0025B 36$:      MOVL    #30,35(R2)              ; 3093
                                0000V 31 0025F          BRW     62$
                          23 A2   1D  D0 00262 37$:      MOVL    #29,35(R2)              ; 3094
                                0000V 31 00266          BRW     62$
                                0000V 31 00269 38$:     BRW     62$
                          50 00000000G  EF  D0 0026C 40$:      MOVL    FDL_BLOCK,R0            ; 3104
                06        00      34  A0  CF 00273          CASEL   52(R0),#0,#6
                                0000V     00278          .DISPL  45$
                                0000V     0027A          .DISPL  41$
                                0000V     0027C          .DISPL  46$
                                0000V     0027E          .DISPL  47$
                                0000V     00280          .DISPL  42$
                                0000V     00282          .DISPL  44$
                                0000V     00284          .DISPL  43$
                                  00V  11 00286          BRB     48$
                          23 A2   0D  D0 00288 41$:      MOVL    #13,35(R2)              ; 3106
                                  00V  11 0028C          BRB     49$
                          23 A2   10  D0 0028E 42$:      MOVL    #16,35(R2)              ; 3107
                                  00V  11 00292          BRB     49$
                          23 A2   12  D0 00294 43$:      MOVL    #18,35(R2)              ; 3108
                                  00V  11 00298          BRB     49$
                          23 A2   11  D0 0029A 44$:      MOVL    #17,35(R2)              ; 3109
                                  00V  11 0029E          BRB     49$
```

```
                                Generated Code

                23  A2              0C   D0 002A0 45$:    MOVL    #12,35(R2)                  ; 3110
                                    00V  11 002A4         BRB     49$
                23  A2              0E   D0 002A6 46$:    MOVL    #14,35(R2)                  ; 3111
                                    00V  11 002AA         BRB     49$
                23  A2              0F   D0 002AC 47$:    MOVL    #15,35(R2)                  ; 3112
                                    00V  11 002B0         BRB     49$
                                         002B2 48$:
                                    00V  11 002B2 49$:    BRB     62$
                    50 00000000G    EF   D0 002B4 50$:    MOVL    FDL_BLOCK,R0                ; 3122
            07      00      34      A0   CF 002BB         CASEL   52(R0),#0,#7
                                0000V        002C0         .DISPL  58$
                                0000V        002C2         .DISPL  55$
                                0000V        002C4         .DISPL  51$
                                0000V        002C6         .DISPL  56$
                                0000V        002C8         .DISPL  52$
                                0000V        002CA         .DISPL  54$
                                0000V        002CC         .DISPL  57$
                                0000V        002CE         .DISPL  53$
                                00V  11 002D0         BRB     59$
                23  A2              23   D0 002D2 51$:    MOVL    #35,35(R2)                  ; 3124
                                    00V  11 002D6         BRB     62$
                23  A2              25   D0 002D8 52$:    MOVL    #37,35(R2)                  ; 3125
                                    00V  11 002DC         BRB     62$
                23  A2              27   D0 002DE 53$:    MOVL    #39,35(R2)                  ; 3126
                                    00V  11 002E2         BRB     62$
                23  A2              26   D0 002E4 54$:    MOVL    #40,35(R2)                  ; 3127
                                    00V  11 002E8         BRB     62$
                23  A2              22   D0 002EA 55$:    MOVL    #34,35(R2)                  ; 3128
                                    00V  11 002EE         BRB     62$
                23  A2              24   D0 002F0 56$:    MOVL    #36,35(R2)                  ; 3129
                                    00V  11 002F4         BRB     62$
                23  A2              26   D0 002F6 57$:    MOVL    #38,35(R2)                  ; 3130
                                    00V  11 002FA         BRB     62$
                23  A2              21   D0 002FC 58$:    MOVL    #33,35(R2)                  ; 3131
                                    00V  11 00300         BRB     62$
                                    00V  11 00302 59$:    BRB     62$
                    50 00000000G    EF   D0 00304 61$:    MOVL    FDL_BLOCK,R0                ; 3141
                23  A2      34      A0   D0 0030B         MOVL    52(R0),35(R2)
                    50 00000000G    EF   D0 00310 62$:    MOVL    FDL_BLOCK,R0                ; 3145
                27  A2      38      A0   D0 00317         MOVL    56(R0),39(R2)
                    50 00000000G    EF   D0 0031C 63$:    MOVL    FDL_BLOCK,R0                ; 3152
                    00V      3C     A0   E9 00323         BLBC    60(R0),65$
                2B  A2              01   90 00327         MOVB    #1,43(R2)                   ; 3154
                                    00V  11 0032B         BRB     66$
                        2B      A2   94   D0 0032D 65$:    CLRB    43(R2)                      ; 3158
                    50 00000000G    EF   D0 00330 66$:    MOVL    FDL_BLOCK,R0                ; 3160
                2C  A2      40      A0   D0 00337         MOVL    64(R0),44(R2)
                    50 00000000G    EF   D0 0033C         MOVL    FDL_BLOCK,R0                ; 3161
                30  A2      48      A0   D0 00343         MOVL    72(R0),48(R2)
                    50 00000000G    EF   D0 00348         MOVL    FDL_BLOCK,R0                ; 3162
                34  A2      4C      A0   D0 0034F         MOVL    76(R0),52(R2)
                    50 00000000G    EF   D0 00354         MOVL    FDL_BLOCK,R0                ; 3163
                38  A2      50      A0   D0 0035B         MOVL    80(R0),56(R2)
                    50 00000000G    EF   D0 00360         MOVL    FDL_BLOCK,R0                ; 3164
                3C  A2      54      A0   D0 00367         MOVL    84(R0),60(R2)
                    50      1E      A2   9A 0036C         MOVZBL  30(R2),R0                   ; 3172
        00000098    8F              50   D1 00370         CMPL    R0,#152
```

```
                                                00V  1E 00377          BGEQU   67$
                    00VFFFFE908   EF                 50   E0 00379          BBS     R0,C.AAJ,71$
                    00V000000006  EF                 00   E1 00381 67$:     BBC     #0,ANALY$IS_ONLY,69$
                                  50          19     A2   9A 00389          MOVZBL  25(R2),R0
                                  10                 50   D1 0038D          CMPL    R0,#16
                                                     00V  1E 00390          BGEQU   69$
                    00VFFFFE903   EF                 50   E0 00392          BBS     R0,C.AAK,71$
                    00V00000000G  EF                 00   E0 0039A 69$:     BBS     #0,ANALY$IS_ONLY,75$
                                  50          19     A2   9A 003A2          MOVZBL  25(R2),R0
                                  10                 50   D1 003A6          CMPL    R0,#16
                                                     00V  1E 003A9          BGEQU   71$
                    00VFFF,F8EC   EF                 50   E0 003AB          BBS     R0,C.AAL,75$
                                  02                 62   91 003B3 71$:     CMPB    (R2),#2                         ; 3181
                                                     00V  12 003B6          BNEQ    73$
                       00000000G  EF  00000000G      EF   D0 003B8          MOVL    DEF_TAIL,DEF_CURRENT            ; 3185
                           0DED   CF                 00   FB 003C3          CALLS   #0,INSERT_AFTER_CURRENT         ; 3186
                                                     00V  11 003C8          BRB     74$
                                      00000001       8F   DF 003CA 73$:     PUSHAL  #1                              ; 3192
                            1189   CF                 01   FB 003D0          CALLS   #1,.NSERT_IN_ORDER
                                                               003D5 74$:
                                                               003D5 75$:
                                  50                 5C   D0 003D5 76$:     MOVL    EDF$LINE_PARSED,R0              ; 3196
                                                          04   003D8          RET

; Routine Size: 985 bytes,    Routine Base: $CODE + 01413

                                                          017EC              .END
```

COMMAND QUALIFIERS

```
  PASCAL/MACHINE/NODEBUG/NOCHECK/LIS=LIS$:EDFUTIL/OBJ=OBJ$:EDFUTIL MSRC$:EDFUTIL

  /CHECK=(NOBOUNDS,NOCASE_SELECTORS,NOOVERFLOW,NOPOINTERS,NOSUBRANGE)
  /DEBUG=(NOSYMBOLS,NOTRACEBACK)
  /ENVIRONMENT= $255$DUA28:[EDF.OBJ]EDFUTIL.PEN;1
  /LIST= $255$DUA28:[EDF.LIS]EDFUTIL.LIS;1
  /OBJECT= $255$DUA28:[EDF.OBJ]EDFUTIL.OBJ;1
  /NOCROSS_REFERENCE /ERROR_LIMIT=30 /NOG_FLOATING /MACHINE_CODE /NOOLD_VERSION /OPTIMIZE /NOSTANDARD /WARNINGS
```

COMPILER INTERNAL TIMING

Phase	Faults	CPU Time	Elapsed Time
Initialization	91	00:00.5	00:03.5
Source Analysis	972	00:21.5	03:10.5
Source Listing	50	00:03.9	00:07.8
Tree Construction	134	00:01.6	00:03.5
Flow Analysis	46	00:00.9	00:02.1
Profit Analysis	51	00:01.2	00:02.9
Context Analysis	479	00:13.8	00:26.7
Name Packing	6	00:00.4	00:00.9
Code Selection	41	00:01.9	00:03.6
Final	437	00:07.0	00:17.4
TOTAL	2311	00:52.9	04:19.1

COMPILATION STATISTICS

```
  CPU Time:       00:52.9         (3627 Lines/Minute)
  Elapsed Time:   04:19.1
  Page Faults:    2311
  Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

EDFTABLES
LIS

EDFTYPE
LIS

EDFUTIL
LIS

EDFTERM
LIS

EDFSTRUCT
LIS

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

EDT

EDT
MAP

EDTVAR
LIS

EDTSHR
MAP

EDT
REQ